A Simulation Performance Study of DTN Algorithms in Practical Use Cases

A Thesis Submitted to the
College of Graduate and Postdoctoral Studies
in Partial Fulfillment of the Requirements
for the degree of Master of Science
in the Department of Computer Science
University of Saskatchewan
Saskatoon

By
Fatemeh Zare Abandankeshi

Permission to Use

In presenting this thesis in partial fulfilment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Computer Science
University of Saskatchewan
176 Thorvaldson Building,
110 Science Place
Saskatoon, Saskatchewan
S7N 5C9
Canada

OR

Dean, College of Graduate and Postdoctoral Studies
University of Saskatchewan
116 Thorvaldson Building,
110 Science Place
Saskatoon, Saskatchewan
S7N 5C9
Canada

Abstract

Delay Tolerant Networks (DTNs) could have substantial value in areas where Internet infrastructure is expensive or dangerous to deploy. A DTN consists of a set of nodes that can transfer messages to each other and immediate packet delivery is not necessary. Pocket Switched Networks (PSNs) are a special case of DTNs where packets are forwarded based on the historical contact patterns between nodes which are assumed to be mobile agents like people or animals. Routing is a challenge in PSNs since an end-to-end path is unlikely to be available from source to the destination.

Previous works proposed the idea of utilizing the social behavior of human contacts to apply different decisions for routing based on social clustering. These ideas can improve PSNs performance in terms of delivery ratio, energy consumption, and delivery delay because transmitting messages around a group is easier due to the higher probability of contact between source and destination.

Contact stability and diversity, network resource capacity, clustering algorithms, and the transmission range of devices may affect the performance of PSN routing algorithms. In this thesis, the effect of each of these parameters on the performance of PSNs algorithms is evaluated by different use case scenarios.

Evaluating the performance of PSN routing algorithms with different circumstances requires a framework that supports cluster-based routing algorithms. Previous DTN simulators do not explicitly support cluster-based routing algorithms. In this thesis, a DTN simulator, PYDTNSIM, has been extended to compare different available cluster-based routing algorithms. This simulator is modular and can be extended for the implementation of other routing and clustering algorithms. Currently, it supports three different clustering algorithms and three routing algorithms (two are cluster-based, and one is unclustered). PYDTNSIM can compare the performance of different PSN routing algorithms in terms of delivery delay, delivery ratio, number of message copies generated, and energy consumption. The simulator can track message transmissions in intermediate nodes, as well as variable message size and several message generation heuristics.

To evaluate the effects of clustering techniques on the performance of PSN routing algorithms, several clustering algorithms are deployed to cluster network nodes. Advanced Graph-based Kmeans (AGKmeans) is proposed in this thesis as a clustering algorithm by using the Kmeans clustering concept. This algorithm is appropriate for datasets in which the participants are dynamic and the dataset can be modeled as a graph. The initial centroids selection in AGkmeans is not performed randomly.

To evaluate PYDTNSIM with different experimental parameters data analysis of the realistic datasets, SHED1 and SHED5, and virtual dataset generation emulator, the Termite, is done to extract the contacts from different environments with different transmission ranges.

Acknowledgements

First and foremost, I deeply thank God for the blessings He has bestowed upon me and for giving me the strength and wisdom to achieve this dream. This master thesis is due to the support and encouragement of many people. It is a pleasure to express my sincere thanks to all those who helped me for the success of this study. I would like to express my sincere gratitude to my supervisors Prof. Dwight Makaroff and Prof. Kevin G. Stanley for the continuous support of my M.Sc. study and related research, for their patience, motivation, and immense knowledge. Their guidance helped me in all the time of research and writing of this thesis.

I would like to thank my lovely Parents, who supported and encouraged me a lot throughout my entire life to achieve my goals. In addition, I would like to thank my dear husband, Ashkan, who supported me spiritually during my married life, especially my M.Sc. study.

Contents

List of Tables

List of Figures

List of Abbreviations

AGKmeans	Advanced Graph-based Kmeans
CBC	Community Betweenness Count
DTN	Delay Tolerant Network
ER	Epidemic Routing
GP	Global Popularity
HCBF	Hierarchical Cluster-Based Forwarding
IPN	Inter-planetary Network
LP	Local Popularity
MANET	Mobile Ad-hoc Network
NCF	Nodal Contribution Factor
PSN	Pocket Switch Network
SHED	Saskatchewan Human Ethology Dataset
TTL	Time To Live
UI	Unique Interaction

Chapter 1

Introduction

On January 28^{th} of 2011, during the Arab Spring, around 80 million people had no access to Internet and cell services in Egypt [58], as a result of disabling both communication modalities by the government to protect itself from groups organized using social media. People were not able to share messages until the government restored the Internet and other communication services. Transferring messages over the pre-existing infrastructure network, such as the WiFi routers and/or cables that operate the Internet, was not be possible. Not only government actions can block the use of communication infrastructure, but also war or natural disasters. These last two scenarios can be even worse than the former because the infrastructure can be damaged and stop working properly in a potentially life-threatening situation. Areas exist where creating infrastructure is dangerous, not possible or expensive [11]. In all these cases, transferring messages without permanent infrastructure could have benefits.

Properly deployed in these cases, a Delay Tolerant Network (DTN) [11] would have a substantial value. A DTN is a communications network where the latency of packet delivery on the order of milliseconds is not critical and eventual delivery is sufficient. In DTNs, nodes can transfer messages to each other via wireless technologies, without needing a pre-existing infrastructure [11]. The network is decentralized, where nodes are responsible for forwarding messages and the message can only be transferred when a node is within the transmission range of another's wireless radio. In a DTN, source and destination nodes might never contact each other directly. Hop-by-hop routing is used, rather than end-to-end routing because of these constraints. Whenever a carrier node of a packet contacts an appropriate intermediate node, based on the policies of current routing algorithms, it forwards the message to that contact node.

At each hop, selecting the appropriate intermediate node can have effects on the usage of bandwidth and buffer space as well as energy, delivery ratio, and delivery delay. Inappropriate intermediate nodes may cause packet loss, wasting bandwidth and buffer space, and creating long delivery delays for packets [47].

There are other use cases where DTN technology can provide communication where standard Internet infrastucture and/or protocols are not available or have poor performance and high cost. Plant monitoring with sensor devices and cameras is a good use case of DTNs which contributes to better environmental monitoring for plant growth, health, irrigation, and productivity. Farmers may wish to receive the environmental data of the field or images of plants without commuting to each location. However, transmitting the plant images or sensor data packets in a large geographic area to the farmers is not conveniently practical over

pre-existing infrastructure networks (such as cellular communication) and creating an infrastructure for this area of farms would be expensive, and using cellular infrastructure would be prohibitively expensive.

The DTN concept is similar to Mobile Ad-hoc Network (MANET), which is an infrastructure-less network of mobile devices that are connected wirelessly [59]. DTN routing is different from MANET routing because end-to-end based routing algorithms designed for MANETs are not capable of routing the packets in DTNs due to the frequent disruption and the sparse contact topology of DTNs [11]. MANET routing is based on transmitting the packets using end-to-end multi-hop routing because this connectivity is available. Due to intermittent connectivity in DTNs, routing is reliant on device mobility and contact opportunity for transmitting messages. Transmission reliability tends to be lower in DTNs than MANETs, since packets may never reach their destinations in DTNs. Table 1.1 compares the differences between MANET and DTN routing, demonstrating that MANET routing algorithms may not always be appropriate for DTNs.

Table 1.1: Difference Between Routing in MANETs and DTNs.

	Routing in Manets	**Routing in DTNs**
End to End Connectivity	Synchronous	Intermittent Connectivity
Delievery Delay	Short	Long
Transmission Reliability	High	Low

1.1 PSN

A Pocket Switched Network (PSN) [34] is a special case of a DTN where packets are forwarded based on historical contact patterns between human users. The word "Pocket" refers to the devices that may be located in a user's pocket. PSNs use contact opportunities to provide human communication in areas where Internet connectivity is not available. As an example, one use case of PSNs is in the villages of developing countries, such as India, Bangladesh, and Cambodia. Because of poor economic conditions, providing permanent infrastructure in the villages of the developing countries may not be possible. In such cases, PSNs can utilize human contact patterns to provide connectivity between source and destination nodes that are geographically separated.

1.2 Challenges and Thesis Motivation

Routing is one of the main challenges in PSNs and in a wider view DTNs. In hop-by-hop routing, selecting the appropriate intermediate node can affect the use of bandwidth, buffer space, and energy. Different studies [35, 57] have investigated routing in PSNs. The results of these studies demonstrate that unclustered routing algorithms cannot gain a high delivery ratio, low energy consumption, and delivery delay simultaneously.

Social behaviour in humans leads to strong contact between groups. By leveraging this idea, transmitting

messages in a group should be easier because the probability of contact between source and destination is higher than transmitting messages between groups [35, 57].

Bubble Rap [35], which is called Bubble is this thesis, used contact features of human social behaviour and proposed the idea of clustering-based on the contacts among participants and then applied different decisions for inter-cluster and intra-clustering routing. Bubble used the local and global popularity of devices for intergroup and intragroup message transmissions. HCBF [57] extended the idea of Bubble and has two different decisions for each of inter-cluster and intra-clustering routing. By considering other factors such as social diversity, and group-betweenness for transmitting messages in each hop, the delivery ratio has been increased and packet loss and redundant message transmission have been decreased in HCBF contact pattern datasets with a random use case scenario [55]. These studies assist PSNs to improve power consumption and delay compared to other routing algorithms. However, there are still challenges and open issues requiring investigation.

One of the main challenges is the difficulty of comparing the performance of various routing algorithms because they are designed for different use case scenarios [11]. Each of these algorithms was applied to different human contact datasets in different environments.

DTN simulators [23, 24, 40] have been implemented to build a single framework for comparison of algorithms. Unfortunately, these simulators do not support the cluster-based routing algorithms that are specially designed for PSNs. The implemented algorithms in these simulators are general routing algorithms such as PRoPHET [45] and Spray-and-Wait [67]. The effect of clustering techniques (*i.e.* community formation) on the performance of PSN routing cannot be investigated easily in these simulators. Algorithms such as Bubble rap [35] and HCBF [57] that are designed for PSNs are difficult to implement and evaluate using these simulators.

1.3 Thesis Statement

The performance of PSNs routing algorithms can be affected by network topology of devices, contacts stability, the transmission range of devices, network resources capacity, clustering algorithms, and message generation rates. Each PSNs routing algorithm reacts differently to the changes in each of these parameters. In this thesis, the dependency of algorithm performance is evaluated by providing experiments in both infinite and constraint resources configurations, over different network topology and contact stability, and simulated transmission ranges. These experiments have been conducted in simulation to quantify the delivery ratio, delay, and energy consumption of each PSN routing algorithms' performance over different use case scenarios. The comparison among the performance of a routing algorithm in different circumstances can determine their applicability.

1.4 Contributions

To evaluate PSN algorithms and represent the performance of different routing algorithms in this thesis, a DTN simulator, PYDTNSIM, has been extended to support routing algorithms using both unclustered and cluster-based heuristics. This simulator is extensible for implementing other routing and clustering algorithms. Currently, PYDTNSIM supports two cluster-based (HCBF and Bubble) and one unclustered (Epidemic) routing algorithms and three different clustering algorithms (Louvain, KClique, and AGKmeans). PYDTNSIM is modular and can be extended easily for the implementation of other clustering or routing algorithms. PYDTNSIM tracks all message transmissions in intermediate nodes, allowing performance comparisons of different DTN routing algorithms in terms of the number of messages generated, the messages delivered to their destination, delivery ratio, delivery delay, number of message copies generated, and energy consumption. Message generators have been implemented for generating variable message sizes with different message generation rates. PYDTNSIM supports variable network resource capacities and different heuristics for determining source and destination pairs. Both unclustered and cluster-based PSN routing algorithms are implemented in a single framework and their performance is compared. The comparison of algorithms is represented using social graphs and histograms over different performance metrics such as delivery ratio, delivery delay, energy consumption, and execution time of the algorithms.

To evaluate the effects of clustering techniques on the performance of PSN routing algorithms, several clustering algorithms are deployed to cluster network nodes for applying different decisions for inter-cluster and intra-cluster packet forwarding. Advanced Graph-based Kmeans (AGKmeans) is proposed in this thesis as a clustering algorithm based on the Kmeans clustering concept. This algorithm is appropriate for datasets in which the participants are dynamic and the triangle inequity cannot be satisfied by participant distances. AGKmeans tries to reduce Dijkstra distances among nodes and their centroid in each cluster. The initial centroids selection in AGKmeans is not performed randomly, centroids are selected based on the number of interaction, unique interactions, and distances to reduce the execution time and increase the reliability of the algorithm.

Analysis of PSN algorithms with different datasets can assist users to understand the effect of network topology on the performance of routing algorithms. Data analysis of actual mobility data is performed to extract contacts from different environments with different transmission ranges. Two datasets (SHED1 and SHED5) with two different methods for contact extractions (WiFi and GPS) with three different transmission ranges have been employed to generate contact graphs. To evaluate algorithms for uses cases that contain both dynamic and static nodes, the Termite Emulator [6] has been utilized to generate several virtual datasets. Three different use case scenarios (random source and destination, plant monitoring, and disaster relief) have been implemented. These scenarios employ different heuristics to determine source and destination for each generated message.

The analysis of results determined the fact that the transmission range of devices has a direct relationship

with PSN routing algorithm performance. Increasing the transmission range of devices catches more contacts, which improves the delivery ratio, average delay, and average energy consumption. Contact stability determined as the duration of each meeting can affect delivery ratio in a way that short-lasting contacts improve delivery. Clustering techniques impact the performance of PSN routing algorithms. Louvain and AGKmeans clustering algorithms have a better delivery ratio than KClique. However, the energy consumption usage of AGKmeans is better than Louvain in most cases. Considering diversity as an additional social factor in HCBF improves the delivery ratio. However, the average energy consumption of HCBF routing algorithms for the sae messages is higher than Bubble. With infinite resources, Epidemic routing has the highest delivery ratio, the best delay among all algorithms. However, Epidemic routing has worse energy consumption. For limited resources, in most cases, single-copy algorithms act the same or better than the Epidemic algorithm.

1.5 Organization of the Thesis

Chapter 2 presents an overview of previous work done on evaluating the performance of DTNs and also a background of the routing and clustering algorithms examed. Chapter 3 focuses on the extension of the DTN simulator which used for evaluating cluster-based routing algorithms that are designed for DTN and also discusses the implementation of AGKmeans clustering algorithm. Chapter 4 provides details of the datasets and the process for extracting the contacts in these datasets for different environments and transmission ranges. This chapter also contains the experimental design descriptions for different use case scenarios. Chapter 5 focuses on the performance of different algorithms under different simulation parameters. Chapter 6 offers discussion, conclusions and the potential scope for future work. This thesis contains two appendices that provide a manual for the download, installation, and implementation of the simulator and emulator that have been used in these experiments.

Chapter 2

Background and Related Works

To understand the research problem of analyzing the performance of PSN algorithms, the chapter provides a background of the implemented algorithms and related works that contain existing DTN simulators and architecture and DTN use cases. DTN routing algorithms can be divided into two different algorithm classes: cluster-based and unclustered. Clustering divides devices into several groups in a way that devices in the same groups are more similar to each other according to a heuristic. Cluster-based routing algorithms apply different routing decisions heuristics to inter-clustering and intra-clustering message forwarding.

Many routing algorithms previously proposed to address PSN routing have challenges because end-to-end connectivity is not available for transmitting the messages. Research on DTN simulators and architecture is also an active area. The simulation section provides a review of DTN simulators along with their weaknesses for the evaluation of DTNs algorithms. The last portion of this chapter describes DTN use cases that are used to inform simulation settings.

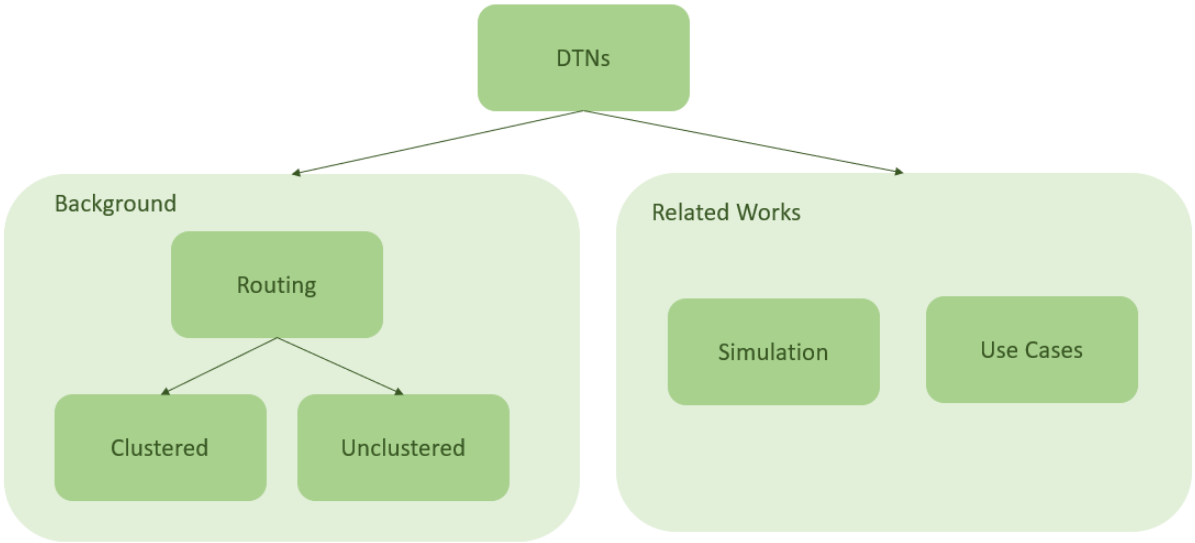


Figure 2.1: Related works schema

2.1 Routing algorithms

Routing algorithms can be divided into Multi-Copy and Single-Copy routing, based on whether there is message replication. This fundamental algorithmic difference is used as the primary organization of routing literature in this thesis document.

2.1.1 Multi-Copy routing

Multi-Copy routing may generate multiple copies of the same message that are routed separately. Increasing the number of copies for each message can improve the probability that one of the copies finds its way to the destination, potentially increasing the delivery ratio and decreasing the delay, at the cost of redundant transmission and storage. For networks where nodes have limited buffer space and the messages have limited time-to-live, multi-copy routing can have a low delivery ratio because packets are dropped due to buffer overflow or message expiry. Epidemic [74], Spray-and-Wait [67], and PRoPHET [45] are classic controlled flooding algorithms following the multi-copy schema. These all have the potential for the destination to receive a message multiple times.

Epidemic routing [74] is a controlled flooding algorithm in multi-copy routing. Epidemic routing nodes continuously transmit messages to new contacts that have not received a copy of the message. Epidemic routing has a guaranteed minimum delivery time. It provides optimal delivery times and reliability if there are unlimited resources. Epidemic routing is used as a baseline to determine the fastest possible delivery time and highest reliability in simulation experiments.

Spray-and-Wait [67] is a multi-copy routing algorithm that limits the number of message copies generated, by applying a maximum number of transmission. The source node that has N copies of the packet would give one copy to other nodes in every contact until the source remains with a single copy. All the carrier nodes that have a single message try to give the message only to the destination node. In binary Spray-and-Wait, half of the copies are forwarded in each contact. This process is repeated by all the carrier nodes until a single copy remains with each carrier node that has been given the messages earlier. The last copy would remain until the node has direct contact with the destination node.

PRoPHET [45], the Probabilistic Routing Protocol using History of Encounters and Transitivity, tries to discover the nodes that have the highest probability of delivering a message to the destination. The PRoPHET algorithm predicts the appropriate carrier for a message by their contact history in the past. In PRoPHET, if contact nodes have a higher delivery prediction than the current carrier of the message, the message would be transmitted to those contact nodes. Thus, one message may be copied more than once, if the current carrier node has more than one contact with higher delivery predictability.

2.1.2 Single-Copy routing

Single-copy routing proposed by Spyropoulos [68], is a simple approach for saving network resources. In single-copy routing, the source delivers the message to an intermediate node that has a higher probability of delivering the message to the destination according to the heuristic used by the particular algorithm. In single-copy routing, each message only has one carrier node in each duty cycle. Overall, single-copy routing has higher latency and lower delivery ratios in comparison to the multi-copy routing algorithms when resource limitations are not a factor. However, single-copy routing saves resources in comparison to multi-copy routing. Each routing algorithm has a policy for selecting the carrier nodes in each hop. In the following, the explanation for several single-copy routing algorithms is provided.

In the Direct Delivery algorithm (DD) [64], the source node would wait for direct contact with the destination node and delivers the message itself. Like the Epidemic routing algorithm, DD is used as a limiting algorithm in simulation studies, representing the minimum energy path, as every successful delivery has a hop count of one.

In the First Contact (FC) algorithm [37], each carrier node delivers the messages to the first node in contact. FC's policy for selecting the carrier node is contact among nodes. Unfortunately, FC suffers from a high delivery delay because the former node that carried the message may have a higher opportunity for meeting the destination node than the current one.

Friendship based routing [8] proposed the idea of utilizing friendship for choosing "next" carrier nodes in DTN routing. The closeness of friendship is determined by behavioural features, such as frequency of contact. Because close friends meet each other more often, they can be a better intermediate node for message routing. Packets would be forwarded to a node if the closeness value of the selected node with the message destination node is greater than the node currently carrying the message.

The SMART routing algorithm [13] provides a social graph for finding the path between the source and destination of a packet. Each node ranks its friends based on the contact frequency, then, nodes exchange these nodes' friends ranking which can create a social graph. The social graph is used to find the shortest path between two nodes.

Gossip [71] proposed a message routing approach, ChitChat, to analyze human activities for finding their social interests. Each person has particular social interests such as shopping, music, or photography. ChitChat creates a social interest profile for each node in a way that while participants meet each other in the same location with a regular period the similarity of their interest will increase. ChitChat forwarding decisions are opportunistic because people usually tend to meet people with similar interests.

The Bubble routing algorithm, proposed by Hui *et al.* [35], considers the popularity of nodes within clusters as a factor for selecting carrier node in each hop. Bubble uses KClique [35] clustering for selecting the carrier nodes of the messages. In Bubble, messages can only pass to nodes with a higher probability of delivery according to popularity metrics. Equations 2.1 and 2.2 represent local popularity and global popularity for each node in the network. In both local popularity and global popularity equations, $g(x, y, k) = 1$ if a contact

between nodes x and y occurs in the kth duty cycle, and 0 if x and y are not in contact in kth duty cycle, where K is the total number of duty cycles. C_x is the set of nodes that are in the same cluster as x. Bubble uses local popularity for intra-community delivery. For delivering the messages to a destination node that is located in the same cluster, a node that has a higher number of contacts in their cluster has a greater probability to be selected as target nodes.

$$\forall(x)LP_x = \sum_{y \in C_x} \sum_{k=0}^{K} g(x, y, k) \tag{2.1}$$

For inter-community delivery, Bubble uses the global popularity of nodes. The nodes which have a higher number of contacts and are not in the same cluster would be selected as target nodes.

$$\forall(x)GP_x = \sum_{y \notin C_x} \sum_{k=0}^{K} g(x, y, k) \tag{2.2}$$

Hybrid Community Based Forwarding (HCBF) [56, 57] has been developed to exploit the Bubble idea by considering the interactions between clusters and social diversity as additional factors in message forwarding. HCBF uses the Louvain clustering algorithm. The interaction between clusters is considered by computing two metrics in HCBF: Community Betweenness Count (CBC) and Nodal Contribution Factor (NCF). Community Betweenness Count is the number of contacts between two clusters and is represented by Equation 2.3. In this Equation, C' defines clusters except C_x, the cluster to which node x belongs. Nodal Contribution Factor (NCF) is a node's contacts with every other cluster which can be calculated using Equation 2.4. CBC and NCF use interactions between clusters as factors for inter-community message forwarding.

$$\forall(C_x, C')_{s.t.(C_x \neq C')} CBC_{C_x, C'} = \sum_{x \in C_x} \sum_{y \in C'} \sum_{k=0}^{K} g(x, y, k) \tag{2.3}$$

$$\forall(x)\forall(C')_{s.t.(x \neq C')} NCF_{x, C'} = \sum_{y \in C'} \sum_{k=0}^{K} g(x, y, k) \tag{2.4}$$

HCBF [56, 57] also utilizes Diversity, which refers to the number of unique nodes that have contact to a particular node, as a factor for intra-community message forwarding. This Diversity metric, Unique Interactions, in HCBF is shown in Equation 2.5. HCBF reduces latency by selecting the most diverse nodes as carrier nodes.

$$\forall(x \in C)UI_x = |S_x| with S_x = \bigcup_{y \in C_x} s.t. \exists g(x, y, k) = 1, k \not< K \tag{2.5}$$

In each duty cycle, carrier nodes may have contacts with a set of other nodes and can transfer messages to these nodes. Message forwarding in HCBF is based on a set of heuristic algorithms. When the carrier has contact with a node that does not belong to its cluster or the destination cluster, HCBF would choose the node with the greatest CBC. When the carrier has contact with a node that is in the same non-destination cluster, the node with the greatest value of NCF would be selected as carrier node. When the carrier node is in the destination cluster, HCBF would choose the node with the greatest UI value as the next carrier. If the UI for several nodes are equal, LP is used.

2.2 Clustering algorithms

As discussed in clustering sections, the specially designed routing algorithms of PSNs such as HCBF and Bubble require a clustering technique to partition the population into groups. Mathematically, clustering is grouping a set of objects in a way that objects in the same cluster are more close to each other than to those in other clusters. Social Network Analysis [62] defined a cluster as a group of people with similar interests that are settled in the same location. However, in PSNs, the definition of clustering is based on the number of contacts. Nodes which meet each other frequently should be placed in the same cluster.

For clustering in graphs, the graph needs to be partitioned into several sub-graphs such that nodes that are more connected are clustered. The fundamental metrics for clustering a graph are the following:

a) Centrality [25, 50] refers to the influence of a node in a graph. In DTN, centrality refers to the social importance of a node. The centrality can be calculated by the number of contacts of each node.

b) The betweenness [25, 27, 62] of a node is based on the number of the shortest paths between nodes in the network that pass through the node. In DTN, nodes with higher betweenness are the good carrier nodes in a network, as it measures increasing contacts among nodes and decreasing the distances among other nodes of a graph.

c) Similarity of graphs can be measured by their nodes similarity which measures how two nodes are similar in the degree [14]. In DTN, the number of common neighbours between two nodes is defined as the similarity of nodes.

Kmeans [48] is a popular unsupervised machine learning algorithm that starts with k random points for k clusters. The algorithm proceeds by alternating between two steps: an assignment step that assigns each node to a cluster based on distance, and an update step that calculates the new means to be the centroids of the new clusters. The algorithm will converge when the assignments no longer change. Kmeans does not guarantee an optimal clustering but is a heuristic that can find locally optimal solutions.

The Louvain algorithm [5] is another heuristic solution that finds clusters with limited predefined information. The Louvain algorithm is fast, simple to implement, and requires limited configuration. The python library, NetworkX, uses the Louvain python package for clustering. The Louvain algorithm works in multiple iterations, each consisting of two phases. During the first phase, each node is considered as a separate cluster. In each iteration, every node is selected and potentially merged with each of its neighbouring clusters to see if the merging improves network modularity. If no potential merges improve the modularity, the algorithm stops. During the second phase, the new community formed in the first phase is reformed to a single node, represented by some centroid value. The phases are repeated until a local near-optimal within some predefined threshold, or a prescribed number of iterations is reached. The algorithm is guaranteed to converge, but may not be optimal, as the heuristic solution uses a greedy search. The advantage of Louvain compared to other well-known clustering algorithms such as Kmeans and KClique clustering is that Louvain does not require advanced knowledge such as the predefined number of clusters.

The Kernighan-Lin (KL) [41] algorithm tries to maximize the number of edges inside a cluster to the number of edges between different clusters. Initial partitions are so important in KL because each step of the algorithm depends on partitioning of in the last step. KL initiates with some partition and repeatedly swap nodes between the partitions to increase the number of edges inside clusters. The nodes would be swap if the number of edges between clusters decreases.

The Girvan–Newman algorithm [27] detects communities by betweenness. Edges with the highest betweenness in the graph are removed. The betweenness of all edges affected by the removal edge is recalculated. These steps are repeated until no edges remain but the algorithm can stop once the number of clusters is reached to a predefined number.

A clique in a graph is a subset of the nodes such that each pair of nodes are connected by an edge. A KClique [19] clustering is defined as the maximal union of all cliques of size k that can be reached through adjacent k-cliques. Two k-cliques are considered adjacent if they share $k-1$ nodes. KClique [52] clustering needs the value of k, the size of the smallest clique, as input. The KClique clustering algorithm could cluster the graph in such a way that only fully connected subgraphs would be found as clusters. Figure 2.2 demonstrates the 3-clique and Figure 2.3 demonstrates 3-clique clusters [18]. As shown in these Figures, every 3 nodes that are the vertices of a 3-clique are located in the same clusters. The pink cluster contains two 3-cliques that one of the vertices has overlap with other clusters. The yellow and blue clusters have five 3-cliques with two and one overlapped vertices, respectively. KClique clustering is not appropriate for sparse graphs because the number of clusters would be high. KClique is not appropriate for dense graphs, because the number of clusters would be one or two.

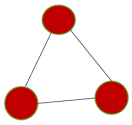


Figure 2.2: 3-clique [19]

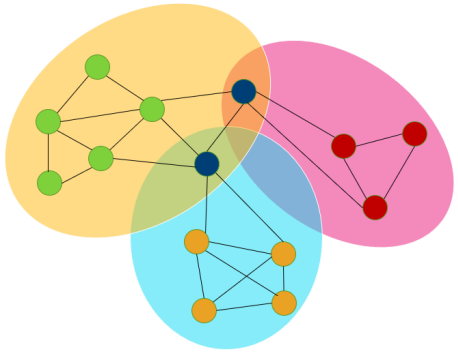


Figure 2.3: 3-clique clusters [19]

2.3 DTN simulators

The simulation and emulation of DTNs are necessary for evaluating new concepts in DTNs such as new routing algorithms. There are several important simulators for DTN routing. NS2 [60] and OMNET++ [33, 75] are general-purpose simulators that have been extended for supporting DTNs [47]. One main challenge with NS2 simulators is limited support for DTN implementation. However, NS2 has different toolsets for MANETs. OMNET++ developed three different mobility models and statistical analysis package which provide the capacity to simulate DTNs. OMNET++ lacks single-copy routing algorithms. NS3 is an open-source C++ network simulator that supports Epidemic [74] and Spray and Wait [67] routing algorithms. [1] Spaho [65] used the NS3 simulator for evaluation of the delivery ratio, hop count, average delay and average buffer occupancy for their DTN use cases. The NS3 simulator has the capability of generating traffic, following node movement, tracing data, and analyzing or visualizing the results. In the remaining part of section 2.3, the simulators which have been developed exclusively for DTNs [47] are discussed.

2.3.1 Opportunistic Networking Environment simulator

The Opportunistic Networking Environment (ONE) simulator [40] is designed for evaluating DTN routing. ONE is very fast and easy to use on both Windows and Linux platforms. ONE supports generating node movement and DTN routing algorithms and has implemented six routing algorithms: Direct Delivery (DD) [64], First Contact (FC) [37], Spray-and-Wait [67], PRoPHET [45], MaxProp [9], and Epidemic routing. These routing algorithms are single-copy, n-copy and multi-copy routing algorithms. The ONE simulator is designed with a modular model. ONE can import real datasets traces and can produce reports from node movement to packet transmission. ONE can visualize both mobility and message passing in its graphical user interface. ONE has limited processing capacity which causes performance reduction on large-scale simulations. The main drawback of the ONE simulator is the lack of cluster-based routing implementations.

2.3.2 DTNRSIM

DTNRSIM [23] evaluates various routing algorithms designed for Delay Tolerant Networks in Java and is a Windows-based simulator. The simulation can present a report of the average data delay suffered, the number of messages generated and delivered during a simulation. DTNRSIM is able to show an event log panel of links creation, delivered and aborted messages detail. Direct Delivery (DD) [64], First Contact (FC) [37], Epidemic, Spray-and-Wait Binary, and Spray-and-Wait Normal [67], PRoPHET [45] and NECTAR [28] have been implemented in DTNRSIM. Message transmission from source to destination is not clear in DTNRSIM. DTNRSIM is not able to compare different DTN routing algorithms on significant metrics such as latency and hop-count. DTNRSIM lacks cluster-based routing algorithms support.

[1] The NS3 Network Simulator, https://www.nsnam.org/

2.3.3 DtnSim

DtnSim [24] is a DTN simulator written in Python, which is specifically designed to evaluate scheduled DTNs but not a suitable simulator for large-scale DTNs. DtnSim's functionalities can be modified and new features can be added. DtnSim modules can generate traffic and decide to transmit a message in each contact or not. DtnSim has no implementation for DTN routing algorithms but DTN routing can plug in via an interface. DtnSim lacks the support of cluster-based routing algorithms that are designed for DTNs.

2.3.4 Urban Delay Tolerant Network Simulator

Urban Delay Tolerant Network Simulator (UDTNSim) [3] is an open-source simulator developed in Python and can simulate urban road networks with different mobility models and routing algorithms of DTNs. UDTNSim implemented three routing algorithms: Epidemic Routing, Superior Only Handoff, and Superior Peer Handoff. UDTNSim simulator is flexible and can accept extensions but it has no implementation for cluster-based routing algorithms.

2.3.5 PyDTN

PyDTN is a DTN simulator in Python and C++ by the University of Maryland.[2] PyDTN uses the Simlpy framework, which provides the event-passing and synchronization and was written for PyDTN. PyDTN has a mobility package that provides a simple mobility model for mobile nodes. In PyDTN, routing tables are used for static nodes message forwarding and Epidemic is used as a routing algorithm for message forwarding in a way that nodes exchange messages as they have contact to the nodes which have not received the packets before. PyDTN does not implement any single-copy routing algorithms and does not have any support for cluster-based routing algorithms.

2.3.6 Challenges with Previous DTN Simulator

Previous simulators build a single framework for the implementation of DTN scenarios. Each of these simulators adds new features for the evaluation of DTNS algorithms from generating traffic to importing real datasets traces. The main drawback of all these simulators is the lack of supports for cluster-based routing algorithms which are specially designed for DTNs and can improve the performance of DTNs.

Tracking packets in intermediate nodes is another important feature that is not available in all DTN's simulators which enables the simulator to evaluate algorithms in different metrics such as delivery ratio, delay energy consumption.

[2] http://users.umiacs.umd.edu/ mmarsh/pydtn/

2.4 DTN Use Cases

DTNs have different use cases from helping humans during disasters to improving harvests conditions on farms [77]. The following section discusses studies which have explored different DTN use case scenarios.

2.4.1 Monitoring Environment and Habitat

One important use case for DTNs is monitoring the environment, wildlife, weather and water quality, and harvests. Collecting of information relevant to agricultural monitoring such as temperature, precipitation, humidity, soil moisture and even images of plants to the farmer can benefit plant growth, health, irrigation, and productivity. Sensors are deployed in farms for monitoring and monitoring crop conditions [51]. Data sharing become possible by letting tractors, farmers, and vehicles to transfer their data by DTNs. Transferring data by such devices can reduce the operational cost in comparison to creating the Internet infrastructure.

Seye [63] used LoRa transmission for herds seasonal migration in Senegal where lack of infrastructure for the Internet does not allow the farmers to share important information such as the geographic location of the equipment or the status of crops. LoRa technology proposed to enable communication services like message transmission, voice messages, the status of water points, geographic location. LoRa is a low-cost ad-hoc DTN that can cover data transmission for wide areas.

The Zebranet project [80] utilized DTNs to study the habits of zebras. Zebranet recorded a zebra's location information using GPS collars in specific duty cycles. Each zebra moves independently in a landscape and when collars distance of one zebra is in the transmission range of others they can exchange information. The researcher can find the location of zebras without needing a permanent and expensive infrastructure.

The Environmental Monitoring in Metropolitan Areas (EMMA) project [61] develops a measurement for the air pollutants in the city in a decentralized architecture deployed with DTNs in the European Union. The project uses public transport vehicles to collect environmental air data.

Monitoring white tail deer can have a great impact on the ecosystem as their population analysis can inform hunting duration. To monitor the status of white tail deer in Ontario, Canada a wireless DTN sensor network has been developed [72]. Several sensor nodes are deployed on the site, then data mules such as helicopters or automobiles are used to gather data from sensor nodes via the DTN.

In a lake/water Quality Monitoring project by the European Union, the quality measurement sensor data transfer messages in a hop-by-hop manner by utilizing DTNs [22] [69]. The data can be transfer from one ship to another when the ship returns to dock.

The GOLDFISH project [76] was proposed to monitor river pollution in developing countries. GOLDFISH has several sensor clusters, with floating WiFi antennas. Sensors sent messages to a gateway near the riverbank, which routes the messages to an Internet server for data aggregation and analysis.

2.4.2 Urban and Remote Areas

A DTN approach can be beneficial and cost-effective for providing Internet services for citizens of urban areas or third world countries, where establishing Internet connectivity is difficult due to economic limitations. Many projects have demonstrated the possibility of Internet access to people in remote villages using DTNs.

KioskNet [29] is a bus-and-kiosk network that developed to provide cost-effective Internet access based on DTNs to people in rural villages. KioskNet utilizes transportations to transfer data between village kiosks and gateways close to urban centers. Wizzy digital in South Africa provides Internet access for some remote village schools. The project provides couriers to drive a motorcycle with a USB storage device and commute between rural schools and cities with a permanent Internet connection to transmit messages [69]. The DakNet project [53] is a low-cost DTN approach that provides low-cost Internet services for rural villages in India and Cambodia. The herders in Padjelanta national park of Sweden had no access to the Internet because the park is a UNESCO World Heritage site and invasive infrastructure cannot be installed. The Saami Network Connectivity (SNC)[44] [46] project proposed the idea of providing the Internet to the herders in remote areas such as Padjelanta national park using DTNs. DieselNet [9] is a project around the urban area of Amherst, MA, USA that contains 40 buses that collect valuable information and transfer them to respective destinations.

DTNs have military applications, modeling collecting battlefield information. Airborne assets such as bombers, tankers, helicopters, and fighter jets play critical roles in communications on a battlefield. DTNs can provide reliable communications among such tools [38].

Automatic health services need innovative technologies for transmitting medical information without the limitation of time and location. DAPHNE [66] utilized DTN approaches for e-Health to transmit messages in rural and urban environments, and the robustness to overcome challenges of disaster areas. The Ghana Consultation Network (GCN)[49] provides electronic health services to improve health care by letting doctors consult and discuss cases with each other. A communication service has been provided based on DTN to about 1700 doctors connected to other members of the medical community by free mobile phone calls and text messages.

2.4.3 Interplanetary communications

Interplanetary Internet can be improved using a DTN architecture and algorithms for creating connections from earth to other planets or spacecraft [43]. The DTN approach can be beneficial for satellite communications as it suffers from long delays and packet losses. Caini [10] proposed the idea of DTN-based communication as an alternative for satellite networking [78]. DTNs are NASA's solution for reliable communication when message transmission from Earth to other spacecraft is a challenge because the distance-based delay and disruption or data loss [36].

2.4.4 Disaster relief

DTNs are suitable communication technology for disaster areas. DTNs can have significant value in disaster relief scenarios when parts of the communication infrastructure are damaged. Uddin [73] proposed a recovery operation for disaster relief through the use of DTNs by establishing a mobility model of population and vehicle movement in disaster areas. During the disaster, citizens are willing to save their device's data such as text, images, audio, and video. DTNs address the challenges of disaster areas such as intermittent connectivity, security and privacy threats [21]. DirMove [30] proposed the idea of the collection of information by mobile nodes from the disaster area and sending of that information to the shelter points by using DTN as communication technology, then, to the control station for analysis. BikeNet [20] is an application for collecting information from bicycle routes. BikeNet uses DTN for regular service and Internet connectivity for urgent message transmission.

2.4.5 Vehicular ad hoc networks

Vehicular ad hoc networks (VANETs) [31] are one of the special cases of MANETs [59], where nodes are vehicles that transfer messages among each other to communicate. VANET's main challenge is the high mobility of the nodes which causes fast topology changes. Unlike DTNs, the end-to-end path between source and destination is sometimes available in VANETs. Packets in DTN usually suffer from long delays compared to VANETs [1]. In DTNs a store-carry forward mechanisms are used for routing to minimize energy consumption for packet transmission in each hop. Different research proposed the idea of vehicular delay-tolerant networks in which VANETs are treated as the DTNs [39]. Messages in VDTNs can be delivered to the destination without an end-to-end path through store-carry-forward mechanisms.

Fastest-Ferry Routing in DTN-Enabled VANET (FFRDV) algorithm [79] is proposed for sparse ad-hoc networks to support communication among vehicles in the highway with high speeds and few traffic lights. In the FFRDV, the roads are classified into the blocks based on geographic locations and vehicles share their current location and speed with other vehicles in the same block. Nodes can communicate the data with other vehicles in their blocks and message carriers are selected based on speed. Thus, if the destination of a packet is not in the same block, the carrier would transfer it to the nodes with the highest speed.

2.5 Summary

Evaluation of DTN routing algorithms needs to be done in a simulation environment which can support cluster-based routing algorithms. Previous DTN simulators had a lack of support for cluster-based routing algorithms which are designed for DTNs. Several cluster-based routing algorithms had been improved the performance of DTNs. Considering the deficiencies of existing simulators, some of the clustering algorithms in the background section have been implemented in an extended simulator to enable the evaluations in

this thesis. Evaluating each routing algorithm with different clustering techniques can determine the effects of clustering on DTN routing performance. Different use cases have been discussed in related works which include a wide range of communication from plant and habitat monitoring to disaster relief. Evaluation of routing algorithms in different use case scenarios can provide more confidences in the analysis of DTN routing algorithms.

Chapter 3

Clustering algorithm and Simulator

Based on the analysis in chapter 2, there are many opportunities to improve the performance of PSNs. Using the social structure of devices to create clusters can enable them to route packets more efficiently in the network. To evaluate PSN cluster-based routing algorithms, several clustering techniques are deployed and designed to evaluate the effect of clustering on routing performance. Advanced graph-based Kmeans (AGKmeans) is proposed as a clustering technique in this thesis for evaluating PSNs routing algorithms. Different cluster-based routing algorithms need to be evaluated in a single framework with the same situation fr a fair and meaningful comparison. A DTN simulator, PYDTNSIM, is introduced and extended to enable the evaluation of PSN cluster-based routing algorithms.

3.1 DTN Simulator: PYDTNSIM

In DTNs, the devices (nodes) are responsible for transmitting the messages from source to destination in each hop. Selecting the appropriate target node affects bandwidth consumption, buffer space used, energy consumption, packet delay, and delivery ratio. To address the challenge of appropriate carrier node selection, previous work [35, 57] in DTNs proposed the idea of clustering network devices and then applying different decisions for intra-clustering and inter-clustering routing decisions. Clustering divides the participants in such a way that the devices in the same group have more contacts or stronger contacts with each other than to those in other groups. Each routing algorithm has a different set of decisions for target node selection that affect the performance of routing in terms of delay, resource consumption, and delivery ratio. Different clustering algorithms affect the routing algorithm's performance because clustering determines the network topology.

It is often difficult to compare the performance of various routing algorithms because they are designed for individual scenarios and with different contact patterns and traffic generation [11]. A single simulation framework for DTNs which can simulate network packet transmission and support clustering would be useful. Previous designed DTN simulators, ONE [40], DTNRSIM [23], DTNSim [24], and UDTNSim [3] lacked DTN routing algorithms such as Bubble and HCBF which employ clustering techniques. A DTN simulator has been designed and implemented, and supports cluster-based PSNs algorithms with variable message size and provides a choice of source and destination nodes to support realistic simulation scenarios.

PYDTNSIM[1] has been implemented in Python as a single framework for DTN algorithms that can support routing algorithms that require clustering algorithms. PYDTNSIM is modular and easily extendable to accommodate other clustering and routing algorithms. The modular-based structure of PYDTNSIM makes it easy to understand and use for experienced programmers. PYDTNSIM evaluates the efficiency of a DTN routing algorithm in terms of delay, the number of packets copies generated, delivery ratio, and energy consumption by recording packets transmissions in intermediate nodes.

3.2 DTN Simulator: PYDTNSIM Architecture

The PYDTNSIM simulator is made of several components. Figure 3.1 demonstrates PYDTNSIM architecture and components.

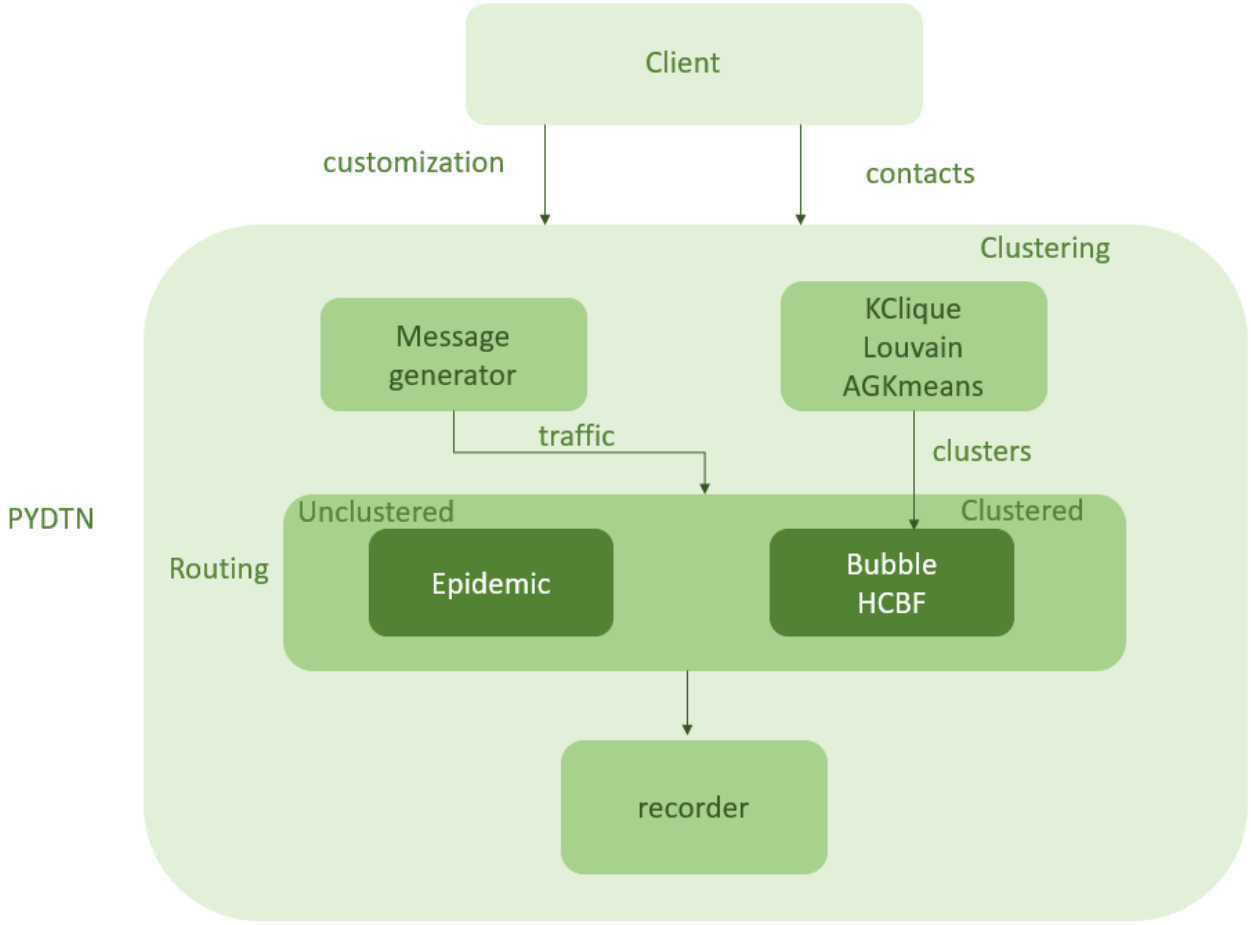


Figure 3.1: PYDTNSIM architecture

PYDTNSIM has defined interfaces for users to import their simulation parameters and contact traces. All

[1] Jarrod Pas and Fatemeh Zare Abandaneshi. https://git.cs.usask.ca/discus/pydtnsim

simulation parameters have default or random values, but users can set parameters for customizing simulation based on their scenarios and import contact datasets. For coordination and communication among different modules, PYDTNSIM uses fixed time steps and different modules communicate with each other during these steps. A message generator has been implemented in PYDTNSIM to generate messages with variable rates.

A clustering module is implemented on top of the routing module. The clustering module accesses the contacts and simulation parameters. Forming clusters based on long-term contact patterns is not appropriate for routing, because social contact patterns may change. Short periods for reclustering are not efficient and would increase the execution time and stability of the routing algorithms. Clusters need to be renewed periodically. This time period is the *epoch* parameter.

General-purpose routing algorithms such as Epidemic only require the message generator output and contact traces for making forwarding decisions. However, cluster-based routing algorithms require clustering results to apply a cluster-based heuristic to routing decision. All packet transmissions in intermediate nodes are recorded, allowing the number of packet copies generated, hop counts and energy consumption for each packet to be measured. When a message reaches its destination, it would be counted as a delivered message and the packet delay and delivery ratio would be updated.

PYDTNSIM contains the following components:

Interfaces for users to import their customization parameters and contact datasets. The *Contact* is defined when nodes meet each other and they can transfer packets. PYDTNSIM has the ability to provide a random choice of contacts, but also lets users import contact datasets because different use case scenarios would have different contact datasets. The contact pattern for a habitat monitoring scenario is different from an Interplanetary communication contact pattern. A PYDTNSIM Input Dataset format has four columns: *time*, *participant a*, *participant b*, and *join*, a binary variable where 1 defines nodes come into contact and 0 defines are no longer in contact. Table 3.1 represents a sample of PYDTNSIM input dataset. Several records of SHED5 that have been transformed into PYDTNSIM format are shown in this table. As the data are collected every 5 minutes in SHED5 the time column is a multiple of 300 seconds. Table 3.1 shows that participant 3 and participant 8 come into contact in time 0 (s) and they are no longer in contact at time 600 (s). Users can import the contact datasets as a CSV file to PYDTNSIM. PYDTNSIM can use contact traces to determine the number of devices/nodes in the network.

PYDTNSIM has the capability of setting *Customization* parameters for each experiment by users. The parameters and their corresponding modules are listed below. All these parameters have a default value if users did not set them.

Algorithm Type: PYDTNSIM modules have been implemented to support both multi-copy or single-copy routing algorithms. Three different routing algorithms (Epidemic, Bubble, and HCBF) with three clustering algorithms (KClique, Louvain, and AGKmeans) are implemented in PYDTNSIM. Although Bubble [35] used KClique and HCBF [57] used Louvain as clustering techniques, PYDTNSIM gives the opportunity to evaluate all combinations of implemented cluster-based routing algorithms with clustering techniques.

Table 3.1: PYDTNSIM Input Datasets Format

time(s)	participant a	participant b	join
0	3	8	1
300	4	8	1
600	4	12	1
600	3	8	0
900	4	12	0
900	5	12	1
1200	5	12	0
1200	4	8	0
1500	4	7	1
1800	4	6	1
1800	4	7	0
2400	4	6	0

By combining the clustering and routing algorithms, PYDTNSIM has seven implemented combination of algorithms: Epidemic, Bubble-AGKmeans, HCBF-AGKmeans, Bubble-Louvain, HCBF-Louvain, Bubble-KClique, and HCBF-KClique.

Step is the message generation rate in seconds (Message generation module).

Epoch determines the reclustering time frame in seconds. By default, reclustering would be done every day (Clustering module).

time-to-live (TTL) determines packet lifetime in seconds. By default, TTL is set to infinity which means the packets will not expire and would remain in the network until the end of the simulation. If TTL is set to an integer value, then the packet would be removed from the buffers of all nodes after that period (in seconds) had elapsed. If the packets reach their destination before their expiry time, they would be counted as a delivered message (Message generation module).

Payload (MBytes) is the packet size. By default packet sizes are set zero (Message generation module).

Bandwidth (MBytes) is the rate of data transmission in each duty cycle is called bandwidth. The users can set this parameter to infinity if they need unlimited data transmission (Routing module).

Buffer Space (MBytes) is the capacity of nodes for storing packets. PYDTNSIM policy for receiving new packets is based on the packets' expiry time (Routing module).

Source and destination of packets. Users can set specific nodes as source or destination of packets. Otherwise, the source and destination of the packets are selected randomly (Message generation module).

seed parameter for random source and destination for different experiments (Message generation module).

Message generator component generates traffic; this is a set of packets/messages for the network. Each generated packet has a defined TTL, payload size, source, and destination node. Users can easily change the

parameters in the *Traffic* function of PYDTNSIM. Packets can have different payload sizes based on their type. Sensor data packets usually have payload sizes smaller than image packets. Users can change TTL and the size of the packets by changing the following parameters. By default, TTL is set to infinity and packet size is zero which defines an unlimited resource scenario.

Clustering may affect the performance of social-based routing like PSNs. Utilizing the social behaviour of humans can improve message transmissions as transmitting messages among people who have frequent contacts like coworkers or family members is easier than people who meet rarely.

Three clustering algorithms are implemented in the PYDTNSIM clustering module: Louvain, AGKmean, and KClique. These clustering algorithms form the clusters for each *epoch*. Different decisions for inter-clustering and intra-clustering can improve PSN routing algorithm performance. By detecting the strength of contacts among nodes from the clustering module, routing algorithms can make better decisions for message forwarding.

While contacts determine the neighbours of each node, the **routing** modules can select the carrier of messages in next hop. PYDTNSIM supports both unclustered and cluster-based routing algorithms. Three different routing algorithms have been implemented in this thesis: 1) Epidemic as an unclustered routing algorithm and 2) Bubble and 3) HCBF as cluster-based routing algorithms. Each routing algorithms has different policies for transmitting packets. In all of these algorithms, each carrier node will check its contacts. For each contact, the carrier node needs to decide whether the contact node is a better carrier for the packet or not, according to the routing algorithm heuristics configured. Each node has a buffer that stores packets in order of arrival. When a node receives a new packet, but does not have enough storage space, a packet must be removed. The default priority for removal is based on the expiry time; nodes prefer to store packets with higher TTL. Figure 3.2 demonstrate a flowchart of packet transmission policies in PYDTNSIM. Epidemic routing is a multi-copy routing algorithm. Bubble and HCBF are single-copy algorithms that work on top of a clustering algorithm. Epidemic routing tries to exchange all the messages until both contact nodes have the same set of messages. For unlimited buffer space and bandwidth, Epidemic results in a maximal delivery ratio. However, if the buffer space and/or bandwidth is limited, Epidemic routing wastes resources because of the high number of copied messages that result in buffer overflow in popular nodes. However, message removal policies can affect the performance of Epidemic algorithms. If a message ends up disappearing from the network, it will never get delivered, and if the message on the fastest path gets delivered, a slower path may still deliver the message.

In cluster-based routing algorithms such as HCBF and Bubble, routing algorithms utilize the output of clustering modules for making forwarding decisions. If nodes are in the same cluster, Bubble [35] uses local popularity for selecting the carrier node of the next hop, otherwise, global popularity would be used. HCBF [57] also utilizes clustering for forwarding decisions. In HCBF forwarding, the cluster of packet destination is an important factor. When a carrier node has contact with a node that does not belong to the carrier node and destination cluster, HCBF decides based on the number of contacts among clusters for forwarding.

When nodes in contact are in the same non-destination cluster the forwarding decision would be based on nodes contact with destination cluster. However, when nodes in contact belong to the same cluster as the destination, the carrier selection is based on diversity. Therefore, the strength of contacts among carrier node, contact node, and the destination of the packet, which is determined by clustering techniques, can change the forwarding decision.

Finally, in each time step of the simulation, the **recorder** module, will record all packet transmissions. Figure 3.3 presents the transition of messages between nodes in PYDTNSIM. When a packet reaches its destination for the first time, the packet will be counted as a received packet. The metrics such as delivery delay, number of hops for delivering packet and energy consumption can be demonstrated for each packet as all transitions are available.

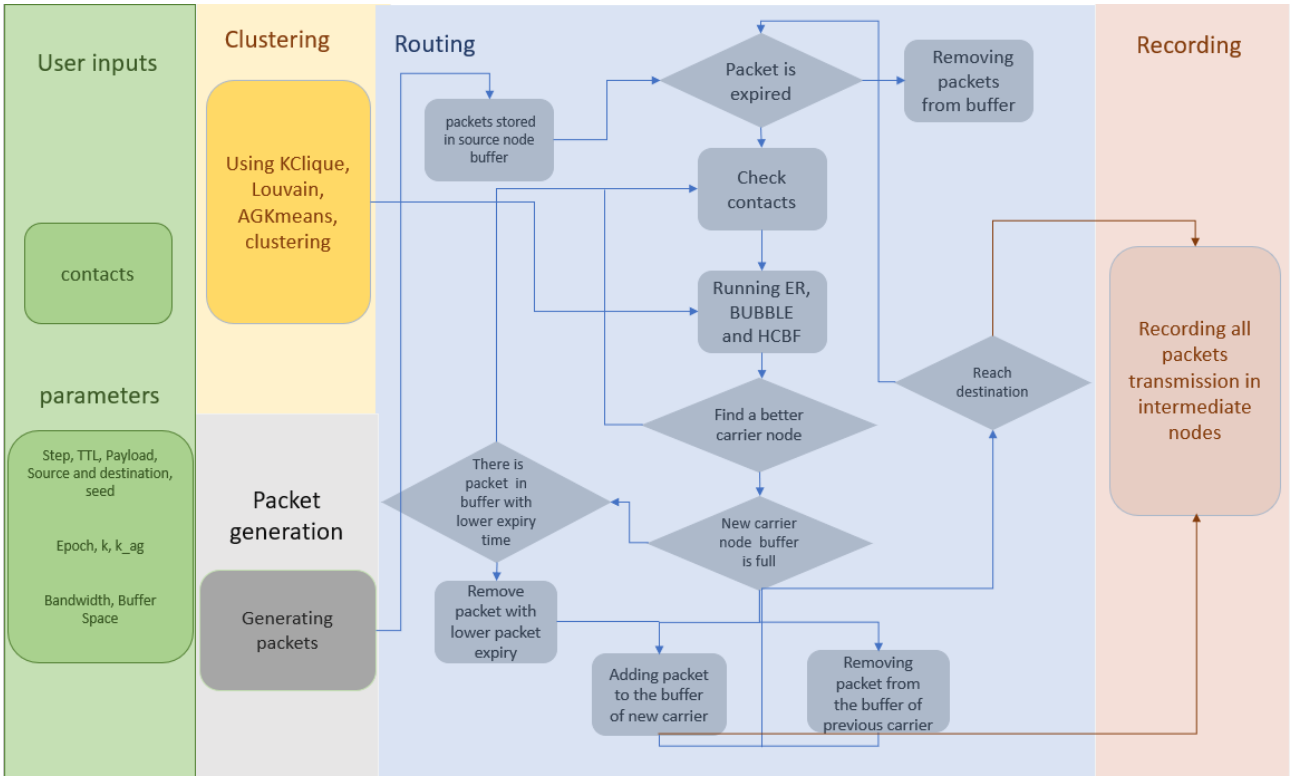


Figure 3.2: PYDTNSIM activity diagram

3.3 Advanced Graph-based K-means

Humans are social beings and they tend to contact people from the same social community such as family, coworkers. This property can be used in human-based routing such as PSNs, where the nodes are devices carried by humans and nodes' contact depends on their movement pattern. Transferring messages in each cluster are typically accomplished faster than transferring messages to other clusters because the probability of direct contact between source and destination is higher within clusters.

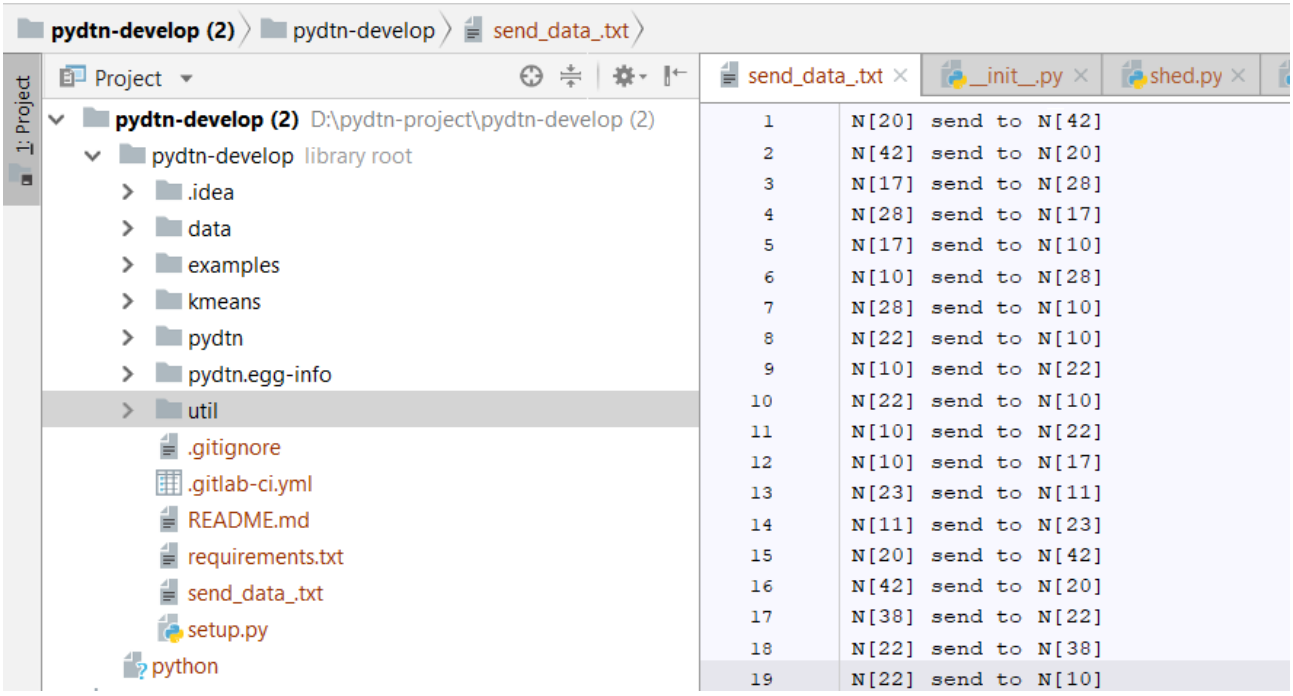


Figure 3.3: Transmission of messages between nodes

It is expected that DTN performance benefits from clustering techniques that capture human social patterns more faithfully. Advanced Graph-based Kmeans clustering is a graph extension of Kmeans clustering. The Kmeans algorithm proposed by Lloyd [48] is a simple iterative algorithm which can partition a dataset into k clusters where k is a predefined number.

The Kmeans algorithm operates on a set of points, D = x_i | i = 1,..., N, where x_i refers the ith point. The algorithm is initialized by selecting k points in D as the initial k clusters which are called **centroids**. Initial centroids are random points. The algorithm iterates among two steps:

Step 1: **Point Assignment**. Each point is assigned to a cluster based on its nearest centroid where the distances between nodes are calculated using a distance metric, which is usually Euclidean.

Step 2: **Centroid Update**. In the update step, the new centroids are selected. These selections are based on the Euclidean distances of centroid adjacent nodes with the nodes in the same cluster. If the summation of this new distance is less than the previous one, the centroid would be updated. K-means iterates between two steps till convergence or stops after a predefined number of iterations.

Since Kmeans utilizes metrics such as Euclidean distance for calculating distances between nodes, these metrics must satisfy the triangle inequality [15]. In clustering human contact datasets that contain dynamic objects for a period of time, the distances among nodes may not conform with the triangle inequality. For example, consider Figure 3.4 which is a student graph that contains three nodes a, b and c. Student a and student b are classmates, and student b and student c are roommates. Thus, a and b will meet each other frequently (2 hours a day) at campus and b and c will meet each other frequently (8 hours a day) in residence.

However, it might be possible that a and c meet each other rarely in shopping center or public transportation (30 minutes a day). In this scenario, triangle inequality is not satisfied. In such situations, classical Kmeans clustering cannot be applied to the problem directly.

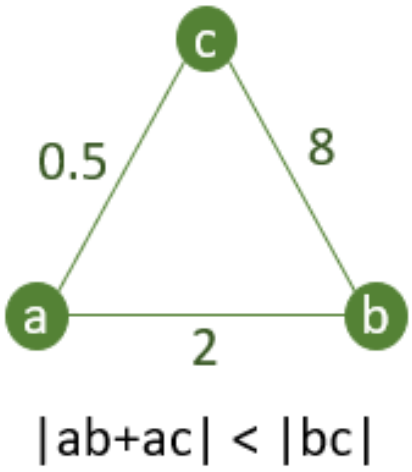


Figure 3.4: triangle inequality neglecting in social-based graph

Both Louvain and Kmeans [48] try to increase the weight of edges in each cluster. However, the main difference is that Kmeans works on data points in a space, while Louvain works on data points connected by a graph, the methodology of increasing the weight of edges in each cluster is different. Louvain tries to increase the weight of edges in clusters by merging sub-graphs. However, Kmeans tries to change the centroids for increasing the weight of edges in each cluster.

3.3.1 Appropriate clustering techniques for social-based algorithms

Clustering is grouping a set of objects in a way that objects in the same cluster are more similar in a factor to each other than to those in other clusters which is called madularity in clustering [70]. As in social-based networks, the distances among nodes cannot satisfy the triangle inequality. In social-based graphs, the weight of edges depicts contact strength among nodes and it is considered that nodes that have stronger contacts have a lower distance to each other. In cluster-based routing algorithms of PSNs, such as HCBF [57] and Bubble [35], both popularity and diversity are utilized as factors for finding appropriate carrier nodes. Consider the popularity and diversity of clustering may increase the performance of the PSNs' routing algorithm. KClique tries to increase nodes' contact diversity in each cluster, and Louvain uses nodes popularity maximization in each cluster.

The stability of algorithms is an important feature for clustering because changing a large number of forwarding decisions in each reclustering is expensive. Nodes need to be informed about the new forwarding policies.

Balancing the number of nodes in clusters is important for using both intra-cluster and inter-cluster

decisions. For example, if one graph has two clusters such that one cluster contains 20 nodes and another has 2 nodes, most of the message forwarding only uses intra-cluster message forwarding. Furthermore, algorithms that cannot make a balance between the number of nodes in each cluster may increase the forwarding responsibility of popular and diverse nodes of each cluster, depleting node resources causing early battery depletion and buffer overflow, increasing packet loss and delay. The execution time of the algorithms is also an important feature for clustering especially when the reclustering period is short and fast cluster results are required.

3.3.2 Advanced Graph-based K-means Concept

Advanced Graph-based Kmeans (AGKmeans) is proposed in this thesis based on a Kmeans clustering algorithm as a new clustering technique. AGKmeans can be applied to datasets that can be modeled as graphs and initial centroid selection is not performed randomly.

In AGKmeans, initial centroids are selected based on the number of unique contacts, number of contacts and distances among nodes. By using the number of contacts and the number of unique nodes contacts, AGKmeans selects nodes with more frequent and diverse contacts as centroids. Nodes with frequent contacts have a shorter distance to their neighbour nodes. Diverse contact nodes have a variety of neighbours. If the initial centroids were not appropriate centroids, their diversity can increase the chance for selecting appropriate nodes as centroids. The selection of nodes with highly diverse contacts as initial centroids can decrease the probability of falling into a weak, but a locally optimal solution. Utilizing the distances among nodes as a factor for initial centroid selection can accelerate the appropriate cluster formation and decrease the number of iterations required to converging to a solution by selecting nodes from different clusters.

The Advanced Graph-based Kmeans algorithm uses the Dijkstra [16] shortest path algorithm. The AGKmeans algorithm has two steps: Assignment step and update step. The algorithm is provided initial centroid nodes as the number of clusters. For initial node selection, nodes would be sorted by their degree and the summation of their edge weights which demonstrate the number of contacts and the number of unique contacts. Then, the distance among the top $2k$ nodes would be calculated by Dijkstra metric and k nodes with the longest distance between them would be selected as initial centroids.

- For the assignment phase, each node would be located in the cluster which have the shortest path length to the node.
- For the update phase, the centroid would be updated in a way to decrease the summation of the shortest path length between the assigned nodes and their centroid in each cluster. The centroids would be updated based on the shortest path to their adjacent nodes. For each centroid, the update would occur if one of the adjacent nodes has a shorter path length to the nodes of that cluster. This method would assist to find the best nodes as centroids which would have the shortest paths to all the nodes of the cluster.

When a new centroid has been found, the algorithm proceeds by switching to assignment step, and this would be repeated up until the algorithm converges to the same cluster collection and no new assignment

changes the shortest path or is stopped by a prescribed number of iterations. Algorithm 1 demonstrates the pseudocode code for the AGKmeans clustering algorithm. AGKmeans has been implemented in python by NetworkX.[2] AGKmeans is implemented as a mudule in clustering module.

Algorithm 1 Advanced Graph-based kmeans clustering

```
Sort nodes based on their degree and the summation of their edge weights
calculate the distances among the first 2k nodes
Insert the first k nodes with longest distance to the centroidlist
assignment(G,centroidlist,df):
for i in centroidnodes: do
  for j in G.nodes: do
    if !(shortest_path_length(i,j)) then
      shortest_path_length(i,j)=inf
    end if
    if shortest_path_length(G,i,j)<df[i][j]) then
      df[i][j]=shortest_path_length(G,i,j)
      shortest_path_length(i,j)=inf
    end if
  end for
end for
update(G,centroidnodes,df):
for i in centroidlist: do
  for j in G.neighbours(i) do
    for k in cluster(i) do
      sump+=shortest_path_length(G,j,k)
      if sump < sum(df[i][k]): then
        i = j
      end if
    end for
  end for
end for
```

Figure 3.5 demonstrates the AGKmeans algorithm in a very simple social graph. In this social graph, the contact strength is shown as labels in some edges and for non-label edges, the weight is considered 1. The algorithm starts with k=2. At first, nodes are sorted based on their degree and the connected edge weights. The top 4 nodes are selected which are shown in part b. Then, the distances among these nodes are calculated

[2] A Python package for the creation, manipulation, and study of the structure of networks https://networkx.github.io/documentation/stable/index.html

and because nodes with labels a and d have the longest distance, these nodes are selected as initial centroids. The assignment phase of AGKmeans is demonstrated in part c, in which two clusters form. In the update phase of the algorithm, node b has been detected as a better centroid instead of a, because the summation of distances of nodes within the cluster to the centroid can improve from 27 to 24. The assignment phase and new cluster formation with centroids b and d are shown in part d. AGKmeans found an appropriate solution in one iteration. However, if nodes a and b have been selected as initial centroid, the algorithm needs more iterations for finding the same solution.

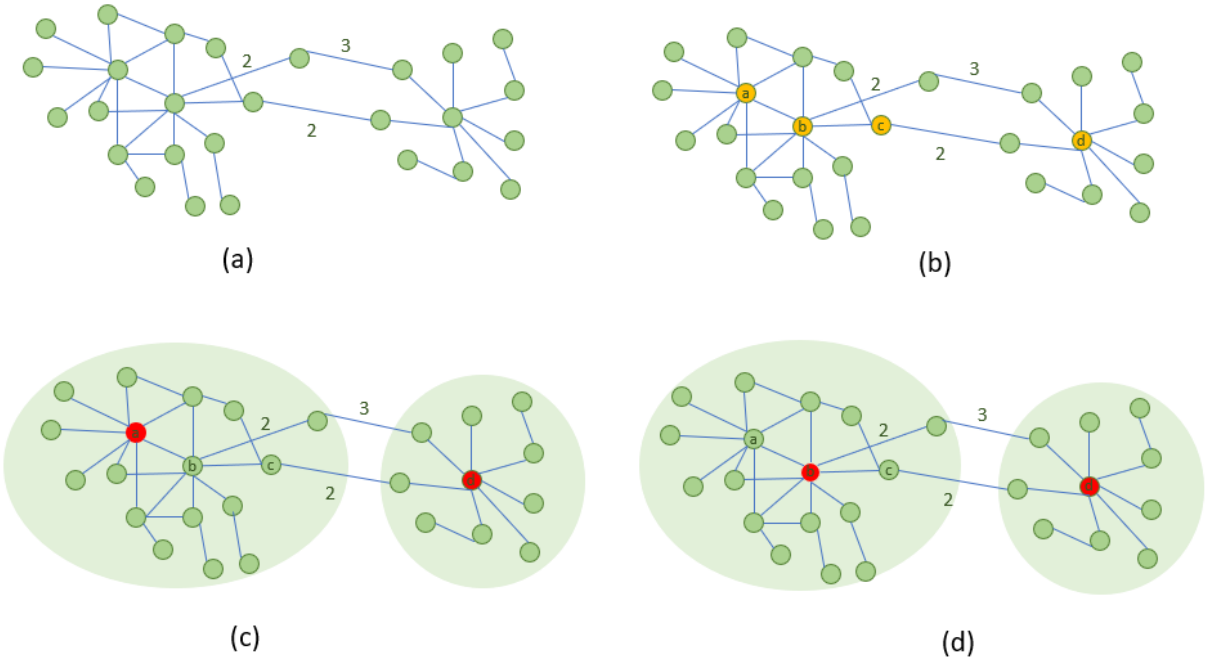


Figure 3.5: Advanced Graph-based kmeans clustering process

3.3.3 Summary

In chapter 3, PYDTNSIM provides a single framework for the evaluation of PSN algorithms. PYDTNSIM supports both unclustered and cluster-based routing algorithms of PSNs. Furthermore, AGKmeans has been developed as a new clustering technique to represent the effect of clustering techniques on the performance of the routing algorithms. PYDTNSIM provides this opportunity to evaluate the implemented algorithms with different contact pattern datasets, network resources capacities, message generation rates, and use case scenarios. In the next chapters, different use case scenarios and contact pattern extraction analyses are proposed and the experimental results with different circumstances by PYDTNSIM simulator would be demonstrated and analyzed.

Chapter 4

Experimental Design

To compare different unclustered and cluster-based routing algorithms in DTNs, several experiments were conducted. Experimental parameters were determined in the context of different datasets, use case scenarios, and participant's device resources. In this chapter, the experimental design including the definition of experimental parameters, specification of performance metrics, the implementation of algorithms, and the process of results visualization are described. The process of generating contact patterns and descriptive statistics on the contact pattern properties for each dataset are also provided.

4.1 Use Case Scenario

As cluster-based routing such as Bubble and HCBF have different policies for inter-cluster and intra-cluster message routing, the source and destination of messages need to be chosen from both of the same groups and different groups to evaluate the performance of each algorithm.

In this thesis, three use case scenarios are defined, as shown in Figure 4.1. These use case scenarios differ in the selection of source and destination node's cluster. While packets' source and destination nodes are from the same cluster, only intra-cluster decisions have an impact on message forwarding and selecting the source and destination of messages from different clusters depends on inter-cluster decisions of routing algorithms. Random source and destination, plant and habitat monitoring, and disaster relief are three applicable use case scenarios that have all possible selection of source and destination nodes cluster and they considered both static and dynamic nodes. In these scenarios, the performance of routing algorithms on dynamic and static datasets using both inter-cluster and intra-cluster message forwarding can be determined.

4.1.1 Use Case Scenario 1: Random Source and Destination

In the random source and destination scenario, the packet generator would select the source and destination for each packet randomly [55], allowing source and destination to be from the same clusters, or from different clusters. All nodes have the same probability for being the source and destination of a packet.

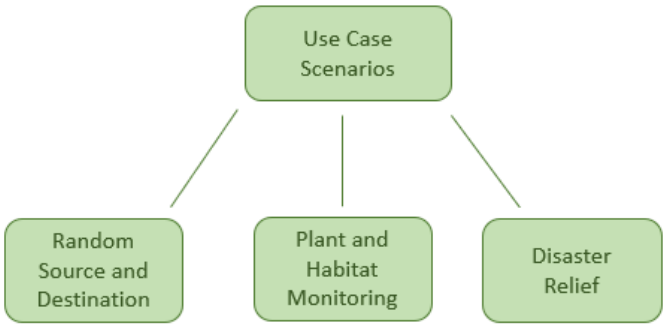


Figure 4.1: Use case scenarios

4.1.2 Use Case Scenario 2: Plant and Habitat Monitoring

Plant monitoring by sensor devices and cameras contributes to having better information on plant growth, health, irrigation, and productivity. Farmers would prefer to receive environmental data from individual plots without commuting to a particular plot. However, transmitting the plant images or sensor data packets over a large geographic area to the farmers is not commercially viable over pre-existing infrastructure. Humans who are passing from farms or near farms can transfer messages to farmers, through a DTN-style system. The plant monitoring network scenario is a combination of static nodes and mobile nodes as shown in Figure 4.2. The antennas represent cameras or sensors which are considered to be static. The dynamic nodes are any tractors, farmers, and vehicles that are passing from farms or near it [63]. SHED1 and SHED5 only contain dynamic nodes. For having a plant monitoring use case dataset, the Termite [6] emulator has been used to generate both static and dynamic nodes. Generating the source and destination of messages in plant and habitat monitoring is performed in a way that either the source or destination is a static node but not both.

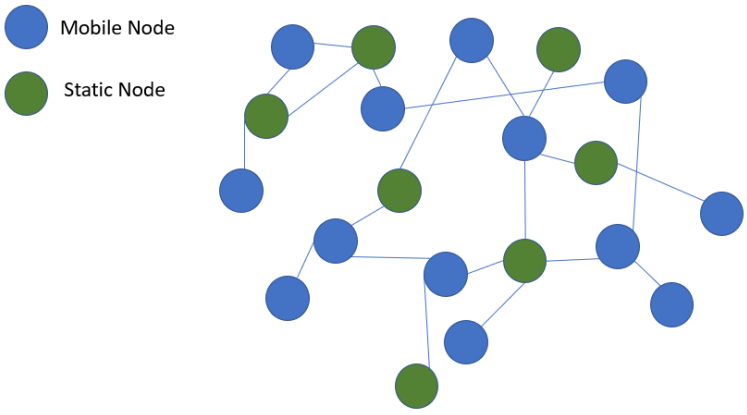


Figure 4.2: Plant and habitat monitoring Use case scenario

4.1.3 Use Case Scenario 3: Disaster Relief

Disaster relief refers to the process of assisting humans in a catastrophic situation caused by some form of disaster. For example, DirMove [30] uses DTNs for collecting information by the mobile nodes from the disaster area and send that information to a center for controlling the situation. For the disaster relief scenario, all nodes are considered mobile, and the disaster area is considered to have limited contact with other nodes as shown in Figure 4.3. Generating the source and destination of messages must be such that the source is in the disaster area and destination of the packet is outside of the disaster area. SHED1 and SHED5 are used for disaster relief scenarios. For generating the source and destination of messages in each run half of the clusters are considered as source and the rest are considered as the destination.

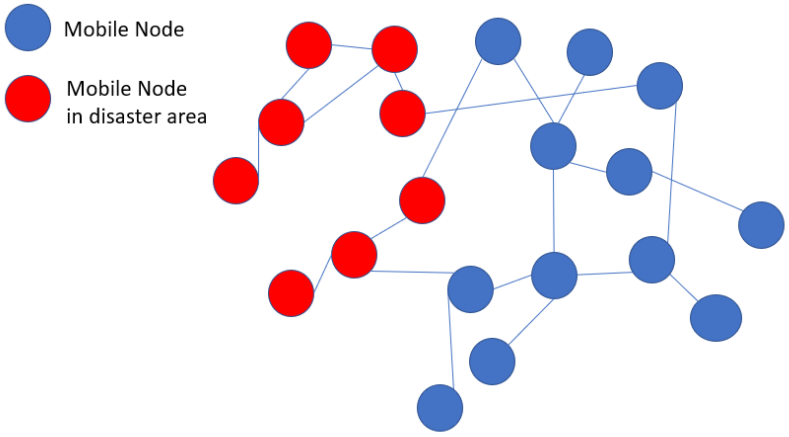


Figure 4.3: Disaster relief Use case scenario

4.2 Datasets

Saskatchewan Human Ethology Dataset (SHED1 and SHED5) [32, 54] contain smartphone-sensor data from students and staff of the University of Saskatchewan (U of S). A summary of the features of the dataset is shown in Table 4.1.

Saskatchewan Human Ethology Datasets were collected through the iEpi [32] mobile app. This app collected the smartphone-sensor data every 5 minutes, a natural choice of the interval for the duty cycle. There are several tables in these datasets. For this thesis, the GPS, WiFi, and battery tables were employed. Table 4.2 and 4.3 demonstrate the tables of SHED1 and SHED5 datasets along with the recorded variables for each table, respectively. With the GPS table, contacts between participants that happened outdoors were recorded, because GPS is not reliable inside large buildings. For indoor contacts, the WiFi table, which gives information about all the WiFi routers that participants have connected with has been used. The battery table shows the battery status of the participant's smartphone. This table was important to select

participants who participated regularly in the SHED study since it contains data for each Duty Cycle that the participant was using the iEpi app. The BlueTooth table in SHED1 demonstrates participant's contact patterns by using Bluetooth proximities among each pair of devices. The acceleration records determine if the phone is not used or left in a fixed place for a significant amount of time when the phone was unplugged [32]. All tables have Participant ID and timestamp to depict the device and the recording time.

Table 4.1: Saskatchewan Human Ethology Dataset details

Dataset	Duration	#(Participants)
SHED1	5 weeks	39
SHED5	4 weeks	29

Table 4.2: SHED1 tables and variables recorded for each table

Table	Variable recorded
GPS	Participant ID, timestamp, Latitude, longitude, velocity, accuracy
WiFi	Participant ID, timestamp, BSSID, SSID, signal strength, frequency, security protocol
Battery	Participant ID, timestamp, Battery level, plugged status, battery status
BlueTooth	Participant ID, timestamp, MAC address, rssi
Acceleration	Participant ID, timestamp, Acceleration in x, y, z

Table 4.3: SHED5 tables and variables recorded for each table

Table	Variable recorded
GPS	Participant ID, timestamp, Latitude, longitude, speed, accuracy, dutyCycle
WiFi	Participant ID, timestamp, BSSID, SSID, capabilities, rssi, frequency, security protocol
Battery	Participant ID, timestamp, BatteryLevel, plugged status, voltage, dutyCycle, battery status, temperature

Table 4.4 represents the number of records for each table of each dataset. From these tables, the contact networks could be determined based on the following Data Analysis methodology, which has different steps to refine the raw data: filtering, stratification, aggregation, and modeling. Figure 4.4 represents Data Analysis methodology steps for two different environments: GPS and WiFi.

Table 4.4: Number of records for each Dataset

Dataset	#(GPS)	#(WiFi)	#(Battery)
SHED1	1,348,034	9,285,061	2,295,104
SHED5	335,941	3,121,983	380,805

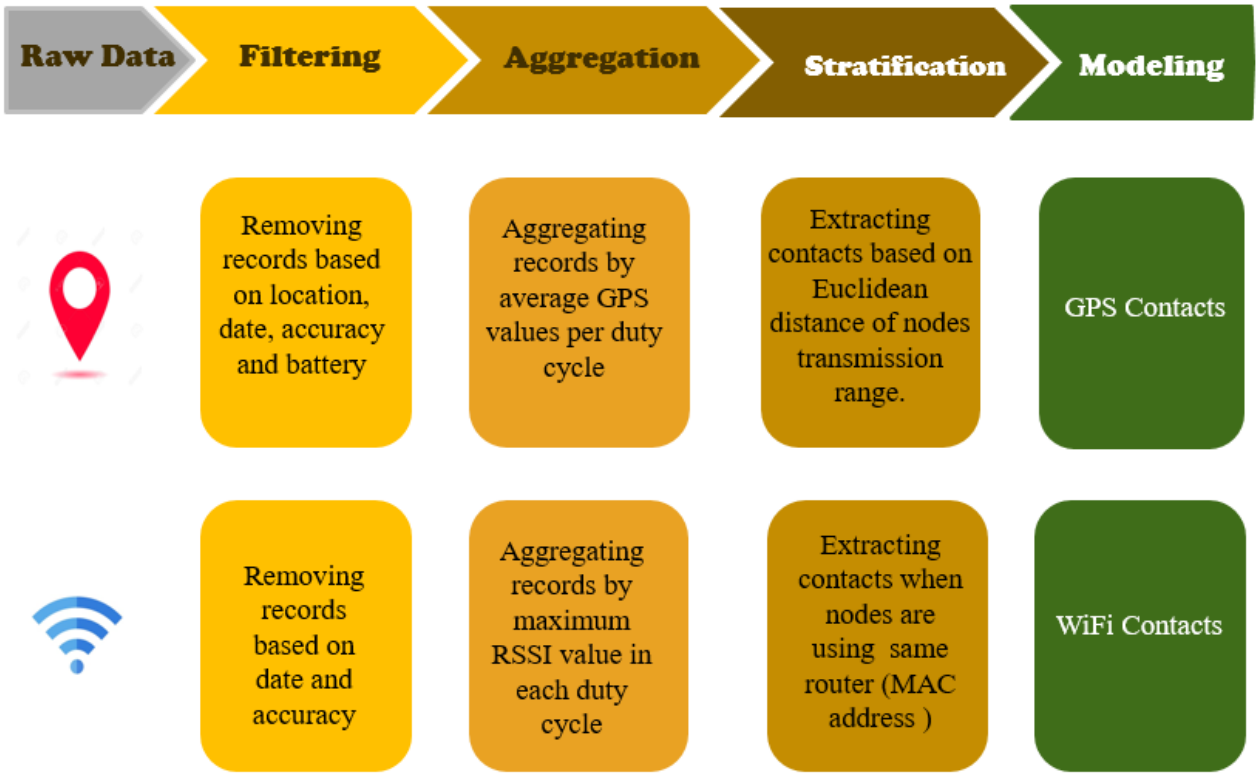


Figure 4.4: Data Analysis Design Pattern

The first step for preparing the data is filtering the unrelated records. In the GPS table, GPS records outside the Saskatoon area, (52.058367, -106.7649138) and (52.214608, -106.52225318), and with an accuracy worse than 100 metres have been removed. For the WiFi table, records with a Received Signal Strength Indication (RSSI) more than -80 dBm are kept. According to experts and websites, (such as MetaGeek [26]), RSSI levels greater than -80 dBm have **good** signal strength. The greater the RSSI level, the closer to the WiFi router the participants are assumed to be. Therefore, good signal strength is important to have some confidence in the participant's location. For both the GPS and WiFi tables, only the records which the participants had less than 50% percent of total possible battery records that can have affect the location accuracy have been removed. Table 4.5 compares filtered GPS and WiFi with the original GPS and WiFi

Table 4.5: Number of records for each Dataset after filtering step

Dataset	#(GPS filtered records)	#(WiFi filtered records)
SHED1	478,949	9,005,061
SHED5	161,891	2,525,661

table.

Since each participant can have more than one GPS or WiFi record in the same Duty Cycle, the filtered data has been aggregated by Duty Cycle and participant. For the GPS table, the first timestamp record and the average latitude and longitude values have been used. For the WiFi table, the maximum RSSI value with its corresponding MAC address and timestamp have been used.

The data has been stratified with different contact definitions. For GPS data, the distance threshold is used, which is the participants' location. Different distances are considered because many technologies with different ranges are available to transfer a message. Three different distances of GPS contacts were considered: 10, 50, and 100 metres. Carettoni [12] classified the Bluetooth technology corresponding to their transmission ranges: 100 and 10 metres. In this thesis, 50 metres was considered to be an approximate mean. For the WiFi table, a contact is considered to have occurred when two participants were maximally connected to the same WiFi router (same MAC address) at the same time. Three different RSSI values corresponding to the outdoor transmission ranges by using the Dong and Dargie's [17] formula:

$$RSSI = -10n \log_{10} d + TxPower, \tag{4.1}$$

where $\boldsymbol{n}$ is the signal propagation constant, $\boldsymbol{d}$ is the distance, and $\boldsymbol{TxPower}$ is the expected **RSSI** value when the participant is within a distance of one meter from the WiFi router were used. The value of n depends on the environmental situation such as wall thickness, number of items in a room and their shapes. $TxPower$ value is more related to the transmitter power, the sensitivity of the receiver. In this thesis, n and $Txpower$ are considered in ranges of 2-4 and 1-7, respectively. The WiFi transmission ranges are calculated based on Equation 4.1 which are -50, -60, and -70 dBm. Table 4.6 represents the number of contacts for GPS and WiFi with different transmission ranges.

The contact extraction method can change the topology of the contact network which may affect the performance of routing algorithms. Figures 4.5 and 4.6 represent two different methods of contact extraction in WiFi and GPS environments which lead to different network topology.

In the GPS environment, contacts are extracted based on the transmission range of devices. While devices are inside the radius of the signal of other nodes, the contact occurs. In this case, the ground interfaces such as walls which can disconnect the connection among nodes that are close enough based on their GPS location are neglected. In this method, the contact network usually generates a partially connected topology with a low number of vertex disjoint subgraphs. In the WiFi environment, devices that are connected to the same WiFi router are known as contacts. In this method of contact extraction, the number of generated vertex

Table 4.6: Number of contacts for each stratification

Dataset	Environment	Transmission Range	#(Contacts)
SHED1	GPS	10	901
	GPS	50	2524
	GPS	100	2707
	WiFi	-50	821
	WiFi	-60	3771
	WiFi	-70	7321
SHED5	GPS	10	1912
	GPS	50	8362
	GPS	100	10448
	WiFi	-50	360
	WiFi	-60	1144
	WiFi	-70	3863

disjoint subgraphs depends on the number of routers in the networks and each distinct subgraph has a fully connected topology.

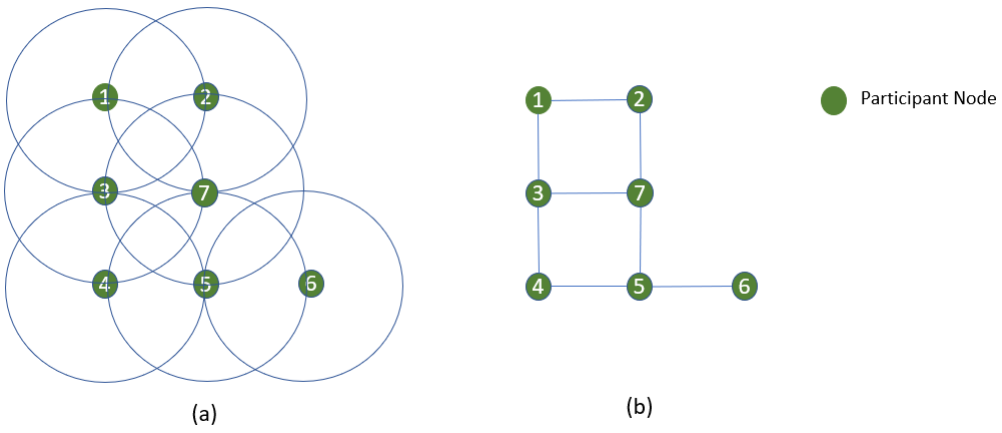


Figure 4.5: GPS contact pattern: (a) the transmission range of each devices (b) extracted contacts

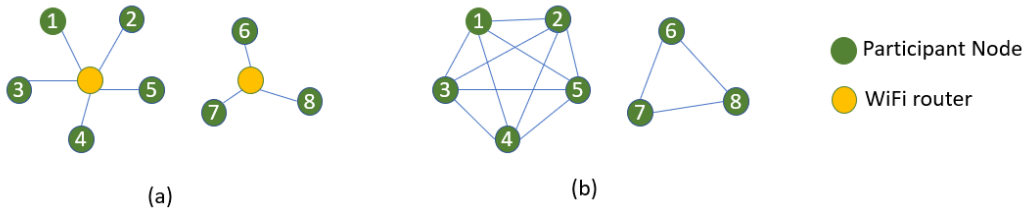


Figure 4.6: WiFi contact pattern: (a) the connection of each devices to the WiFi router (b) extracted contacts

4.2.1 Termite dataset

The Saskatchewan Human Ethology Datasets such as SHED1 and SHED5 [32] are not sufficient for use case scenarios which have a mixture of static and dynamic devices, or scenarios which are not based on the contact patterns of university students. As mentioned in the use case scenarios section, in scenarios such as plant and habitat monitoring both static and dynamic devices are responsible for transferring packets in the network. The antennas placed in farms as sink nodes for gathering cameras and sensor devices data which considered to be static. To have a dataset similar to the plant and habitat monitoring dataset which has both static and dynamic nodes, an emulator has been used to generate a dataset that contains both static and dynamic nodes.

Termite [6] is a software emulation testbed that enables Android developers to run applications on virtual contact networks. It provides a testbed that developers can define a virtual contact network including its topology, how it grows over time with different sizes and dynamic objects. Termite datasets are generated with dynamic devices where each device emulates the features of a real Android device. Appendix B provides a short tutorial for installation and running Termite.

Termite starts by producing a predefined number of devices in a predefined area. Each device has a random walk movement [4] and in each duty cycle, it decides either to move or to stay. This decision is based on a probability that is given to each node as the movement pattern. For each move, each device can walk with a predefined distance in any direction, forward, backward, left or right. When two or more devices are close enough based on their signal strength the contact among them would be detected. Each device has a location and signal strength which differ from other devices.

Figure 4.7 demonstrates the setting of a typical Termite device which is Andorid devices such Nexus S and Nexus 1. The location and the signal strength of each device can be customized by users. For generating the plant and habitat monitoring dataset, 30 android devices are defined in an area of $10km^2$, and 9 nodes are set to static by selecting their probability of movement to zero and the rest of devices are dynamic. Static nodes are located in a way that the longest path among each pair of devices is two hops. For dynamic nodes, in each duty cycle, nodes can remain in their previous location or move $10m$ forward, backward, right or left with equal probability for each direction. These 30 devices are selected randomly from one the devices

provided by Termite. The considerations have been run 5 times and result in 5 different datasets. The results demonstrated in the following sections are the average for these 5 datasets.

Based on Figure 4.8, SHED1 and SHED5 and Termite are the datasets that are chosen for the experiments. Each of SHED1 and SHED5 datasets have different environments, GPS and WiFi. As limiting the transmission ranges of devices can change the number of contacts, three different ranges have been considered. Termite datasets are used for Plant and habitat monitoring. The random use case and disaster relief scenarios use SHED1 and SHED5 as their datasets.

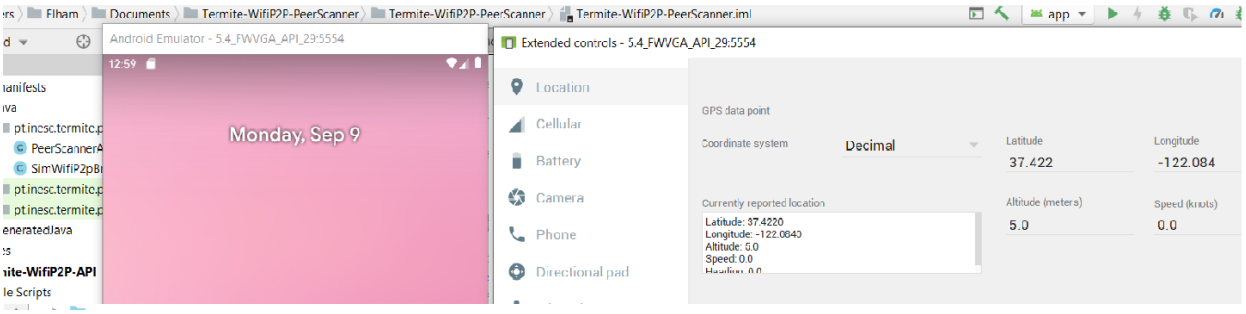


Figure 4.7: Termite device settings

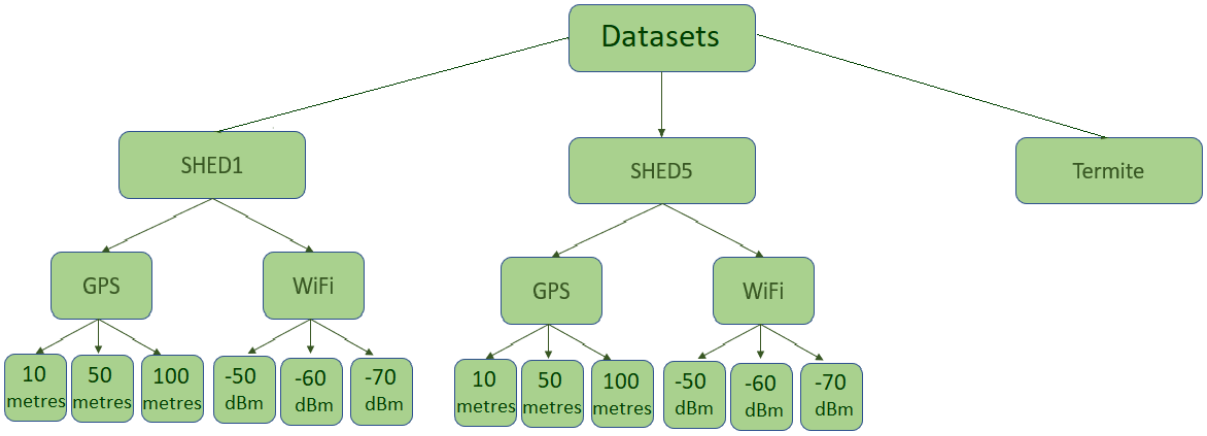


Figure 4.8: Datasets for designing experiments

4.3 Participant Device Resource Capacity

To compare PSNs algorithms for each scenario, one set of experiments are executed to demonstrate how the algorithms performed without resource constraints while another set of experiments addressed more limited resource scenarios where the buffer space and bandwidth are constrained.

Infinite resource scenarios provide a baseline for comparison of other algorithms' performance. Epidemic routing algorithms with unlimited resources capacity provides the best case of delivery ratio and delay among

all available algorithms of PSNs which can be useful for comparison.

For the infinite resource scenario, a node's buffer space is considered unlimited and time-to-live is infinity and packets are zero Bytes. With infinite resources, each contact device can transfer all the eligible messages to other devices, and each node can keep all encountered. For limited resource scenarios, the buffer spaces of nodes have limited space for storing packets. The generated packets would be expired after a finite time, and each packet has a specific payload size.

In HCBF [55] and Bubble [35] experiments, the number of messages generated in each scenario is based on simulation time and buffer capacity was set to a fixed percentage of the number of messages generated. As in PYDTNSIM message generation is flexible and can follow different distributions, the buffer capacity is fixed by message sizes. The message sizes are set such that each buffer space can store 10 messages which are proportional to the buffer capacity set by message generated in Bubble and HCBF. The time-to-live of the packet is 7 days based on HCBF experimental parameters [57].

As message generation in PYDTNSIM is flexible, users can set the message generators to generate based on a defined distribution. In this thesis, uniform and power function distribution are used for message generation. The first message generation follows a uniform distribution that has been used in HCBF [57]. In uniform distribution, one message is generated in each duty cycle. Since the delivery ratio can only measure the packets which have been delivered to their destination in the duration of experiments, in-progress messages at the end of simulation can make issues for delivery ratio. In-progress messages have been generated before, but these packets had insufficient time for being delivered. This has a negative effect on delivery ratio, because the delivery ratio is the ratio of messages received by their destinations to the generated messages. A power function distribution has been used to address this problem. In uniform message generation distribution, the message generators were generating one message per duty cycle from the start of the simulation to the end.

A power function distribution [2] for GPS with the range of 10 metres is shown in Figure 4.9. The shape parameters of power function distribution are set to $\alpha = 0.7$ and boundary parameters are set to the simulation time. In this case $a = 1$ and $b = 8927$ where a is the start duty cycle and b is the end duty cycle of simulation. As shown in this figure, power function will generate more messages at the beginning of the simulation and the message generation rate will decrease by time passing which can reduce the number of in-progress messages at the end of the simulation.

The uniform distribution generates one message per duty cycle, but the power function generates one message per four duty cycle at the start of the simulation and the generation rates decrease such that at the end of simulation for the final 10 duty cycles, only one message is generated. The number of messages generated in the uniform distribution of GPS 10 meter is 8868. In power function, the number of messages generated reduced to 1445.

Table 4.7 demonstrates the resource parameters set up for each use case scenario. In each scenario, network resources capacity, datasets, or message generation rate have been changed to give a better view of the impact of each of them on performance metrics.

Table 4.7: Parameters set

Use case scenario	Message generation distribution	TTL	Buffer Space	Dataset
Infinite resource for Random Src/Des	Uniform distribution with one message per DC	infinite	infinite	SHED1 SHED5
Limited resource for Random Src/Des	Uniform distribution with one message per DC	7 Days	capacity for 10 messages	SHED1 SHED5
Limited resource for Plant and habitat monitoring	Uniform distribution with one message per DC	7 Days	capacity for 10 messages	Termite dataset
Limited resource for disater relief	Power function distribution with $\alpha = 0.7$, $a = 1$, and $b = 1445$	7 Days	capacity for 10 messages	SHED1 SHED5

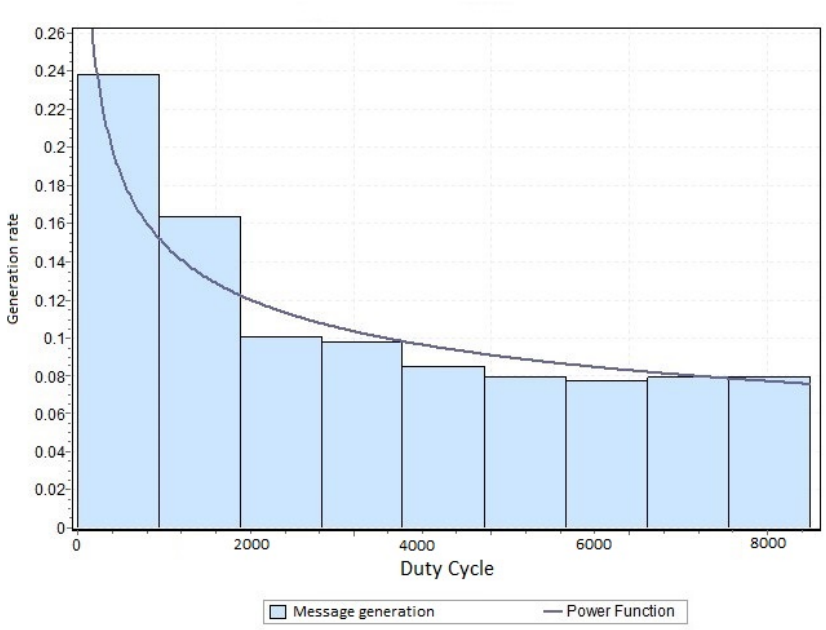


Figure 4.9: Message generation follows a Power distribution

4.4 DTNs evaluation metrics

Routing in DTNs should manage a tradeoff between maximizing reliability such as delivery ratio and minimizing the expenses such as delivery delay and energy consumption. The ideal case of delivering the message

before its presented expiry time with the lowest cost is to keep this message until the destination is in transmission range. The most effective approach to maximize the message delivery ratio and minimize the delivery delay is to forward this message to all other nodes at each contact. It is assumed that DTNs are naturally tolerant of the long delivery delay but would profit from a shorter delivery delay [57]. The evaluation metrics evaluated here are Delivery ratio, Delivery Delay, and Estimated Energy Consumption.

Delivery Ratio: Delivery ratio is defined as the ratio of messages received by their destinations to those generated by the message generators. Each algorithm tries to deliver a higher number of messages to their destination using their particular policies for forwarding messages. One issue with the delivery ratio is in-progress messages at the end of the simulation. These messages have been generated but do not have enough time to be delivered. In multi-copy routing algorithms, only the first copies that reach the destination is counted for delivery ratio.

Delivery Delay: Delivery Delay is the time duration in duty cycles between the generation and delivery of the message. Delivery Delay includes all possible delays such as buffering latency or queuing delay. This metric is calculated by subtracting the time at which the packet was generated by the message generator from the time at which the packet reached its destination. In multi-copy routing algorithms, the delivery delay is calculated based on first packet delivery. Average, maximum, median, and variation in delivery delay are calculated.

Energy Consumption: Energy Consumption in DTNs is usually related to the transfer and reception of messages rather than the computation in each node. In this thesis, the energy consumption is calculated by the number of transmissions.

Execution Time: The Execution Time of an algorithm can presents how quickly the algorithm can be run and use the resources of the compiling machine [42]. The Execution Time of an algorithm is the total amount of time needed by an algorithm to complete and is calculated in this thesis by measuring the start time and end time of each algorithm.

4.5 Result Demonstration/Visualization

The social graph is a common method to represent connections among objects [55]. In this thesis, social graphs are utilized to demonstrate the strength of contacts, the number of unique contacts, clustering techniques performance and the number of messages that each node is carrying in different routing algorithms which can estimate the energy consumption. In social graph visualizations, participants are depicted as nodes in the graph. The strength of contacts between two participants represented by the thickness of the edge. The node size is proportional to the total number of packets forwarded by that node. Nodes with the same colour are located in the same clusters. The thickness of an edge is proportional to the number of contacts between two nodes, the colours represent either the contacts are among nodes in the same cluster or two different

clusters. Graphs are visualized with a social network analysis tool Gephi.[1]

A histogram represents the organization of numerical data. Histograms can depict a comparison of delivery ratio and energy consumption for different algorithms. However, a boxplot can represent the distribution of data by presenting a view of maximum and minimum, median and quartiles of data. The top and the bottom of the box represent the first and the third quartile, whereas the line in the box represents the median.[2] Median is the middle value in a data sample that can separate the higher half from the lower half. The first and the third quartile are the median of the upper and lower half of the dataset, respectively.

4.6 Chapter Summary

The experimental design chapter discussed the experimental set up for the evaluation of implemented algorithms in PYDTNSIM. Defining different datasets and different methods for contact extraction provides different network topologies and number of contacts, which may affect the performance of routing algorithms. Selecting source and destination from both the same cluster and different clusters can demonstrate the effect of social behavior factors in inter-cluster and intra-cluster message forwarding decisions and can demonstrate which social behavior has a better performance on each network topology. Message generation rate can affect the performance of the routing algorithms which can be analyzed by having different message generation rates. Unlimited network resource capacity can provide a baseline for the performance of routing algorithms, realistic resource capacities and message sizes can demonstrate a better comparison among routing algorithms.

[1] https://gephi.org/
[2] https://towardsdatascience.com/understanding-boxplots-5e2df7bcbd51

Chapter 5

Results

The main objective of this thesis is to determine how the performance of PSN routing algorithms may be affected by the topology of devices, contact stability, the transmission range of devices, network resource capacity, clustering algorithms, and message generation rates. The topology of devices, contact stability and the transmission range of devices are derived from the datasets that are used for the experiment. Clustering techniques, variable message generation, and network resource capacity need to be supported by the simulation environment. The previous chapters presented the PYDTNSIM simulator which implemented DTNs routing algorithms. In contrast to the previous DTN simulators, PYDTNSIM can support cluster-based routing algorithms along with variable message sizes and network resource capacities. In this chapter, the features of datasets and their effect on PSNs routing algorithms are presented. Each PSN routing algorithm is evaluated with different network topology, contact stability, transmission range, network resource capacity, clustering algorithm, and message generation rate to compare performance metrics such as delivery ratio, delay, and energy consumption.

5.1 Dataset contact duration

As PSN routing algorithms use contact opportunities for message forwarding, contact features can affect the performance of PSN routing algorithms. In this section, contact duration in each meeting is investigated. Contact duration can help determine the stability of contacts in GPS and WiFi environments based on the measurements obtained in the datasets.

Figures 5.1 and 5.2 demonstrate the contact duration frequency for each pairwise encounter. For example, Figure 5.1(a) shows that all of participant's determined using GPS from SHED1 with transmission ranges less than 10 metres lasted for fewer than 10 duty cycles. However, in Figure 5.1(d), several WiFi contacts with transmission range greater than -50 dBm last for at least 20 duty cycles in SHED1. In Figure 5.2(f), WiFi environments of SHED5 with RSSI greater than -60 and -70 dBm, several contacts last for more than 10 duty cycles are observed, however, in Figure 5.2(c) in the GPS environment, the maximum contact duration is 4 duty cycles.

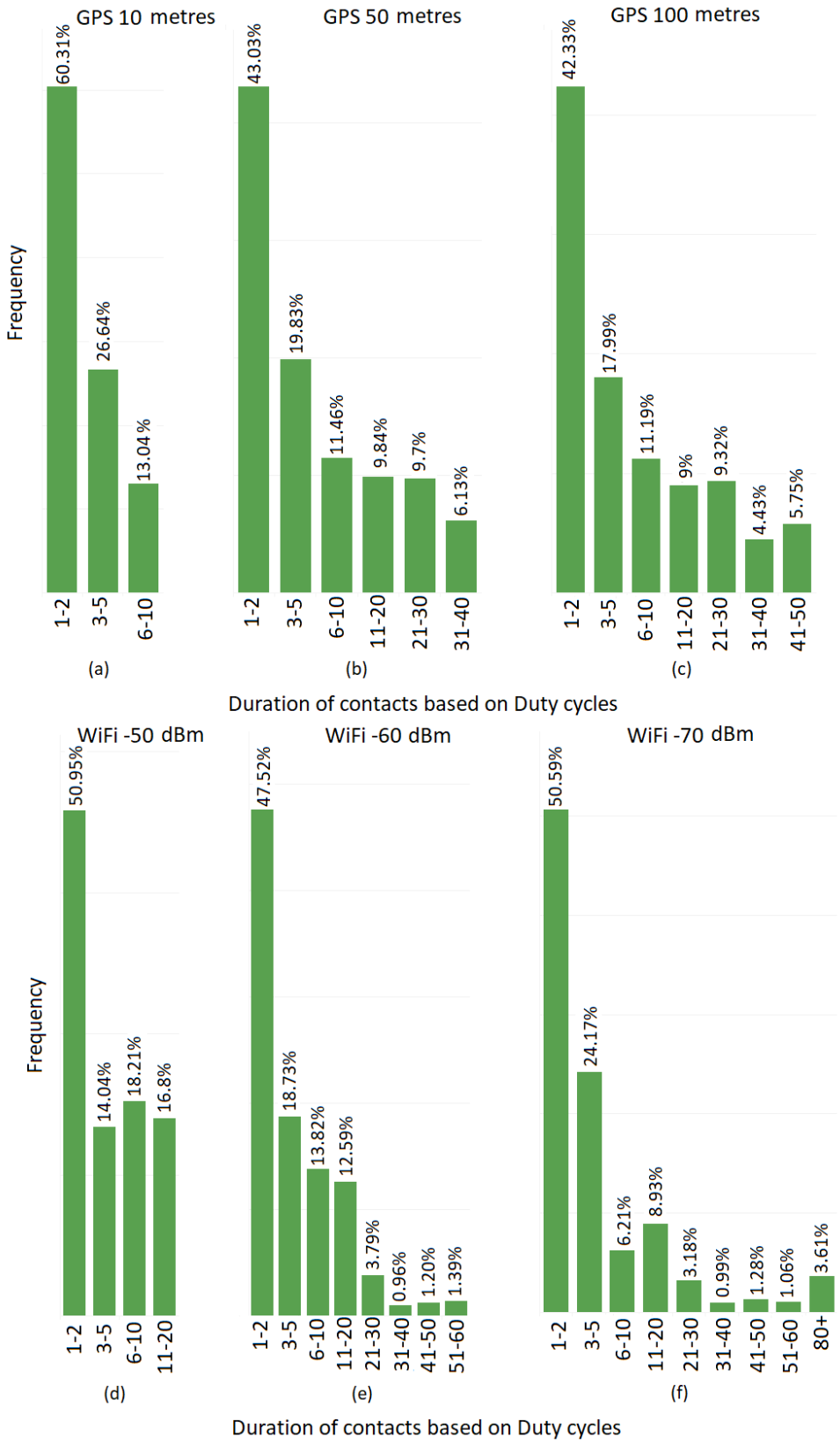


Figure 5.1: Contact Frequency based on duty cycles - SHED1

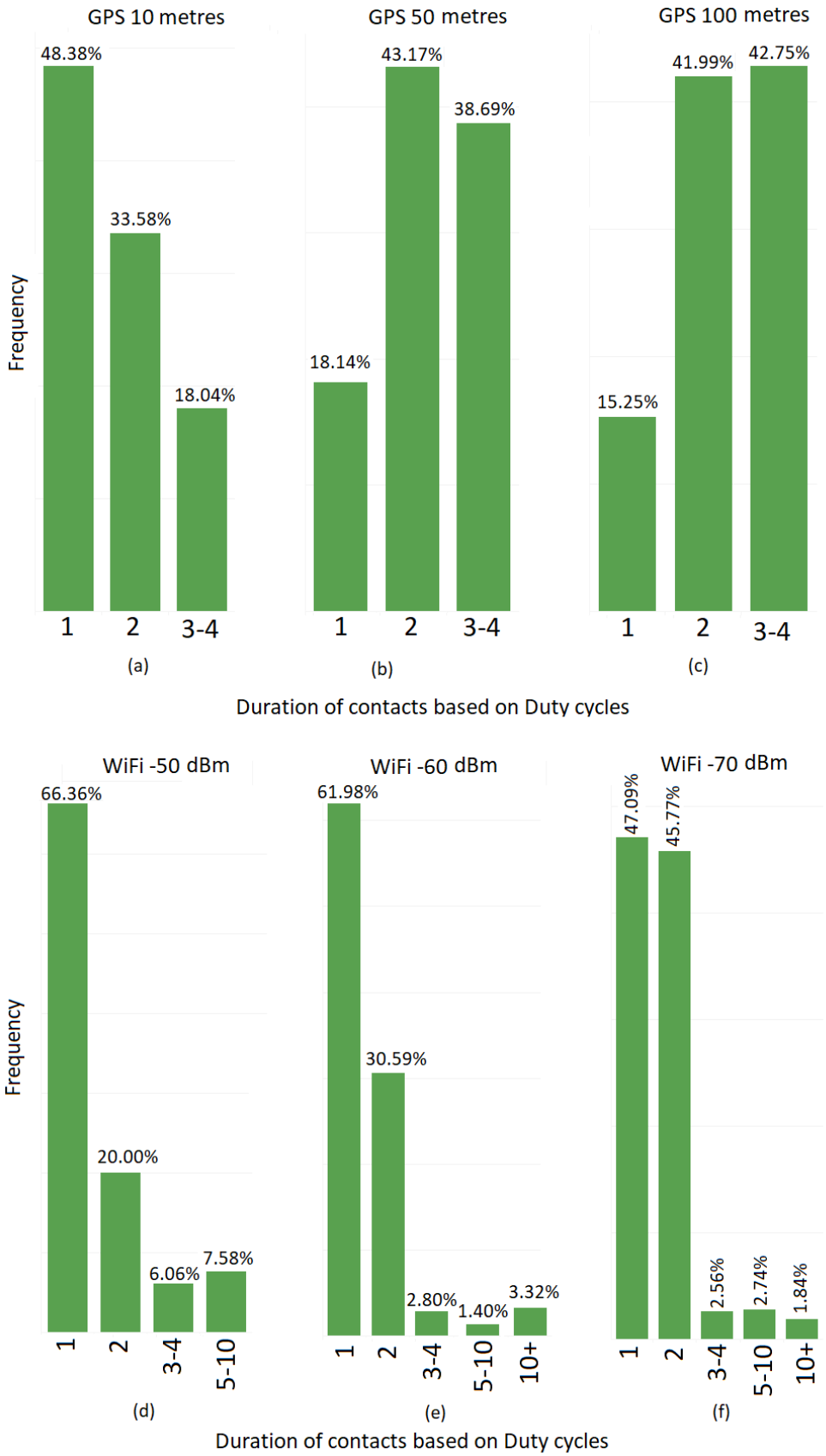


Figure 5.2: Contact Frequency based on duty cycles - SHED5

These figures demonstrate that WiFi contacts are more long-lasting than GPS contacts in SHED1 and SHED5, likely because GPS works better in outdoor areas, as signals are transmitted over waves at a frequency that does not penetrate solid objects such as walls easily and makes it difficult for the device to determine the location accurately [7]. WiFi works better in indoor areas as WiFi routers are not usually available in outdoors. For SHED1 and SHED5 datasets, in outdoor spaces, people are less likely to be stationary as most of the participants are students and staff of the university and will spend most of their time in indoor offices with relatively little mobility. Staying a long time in indoor offices, the WiFi contacts are more long-lasting. Participants' mobility in outdoors causes short-lasting contacts.

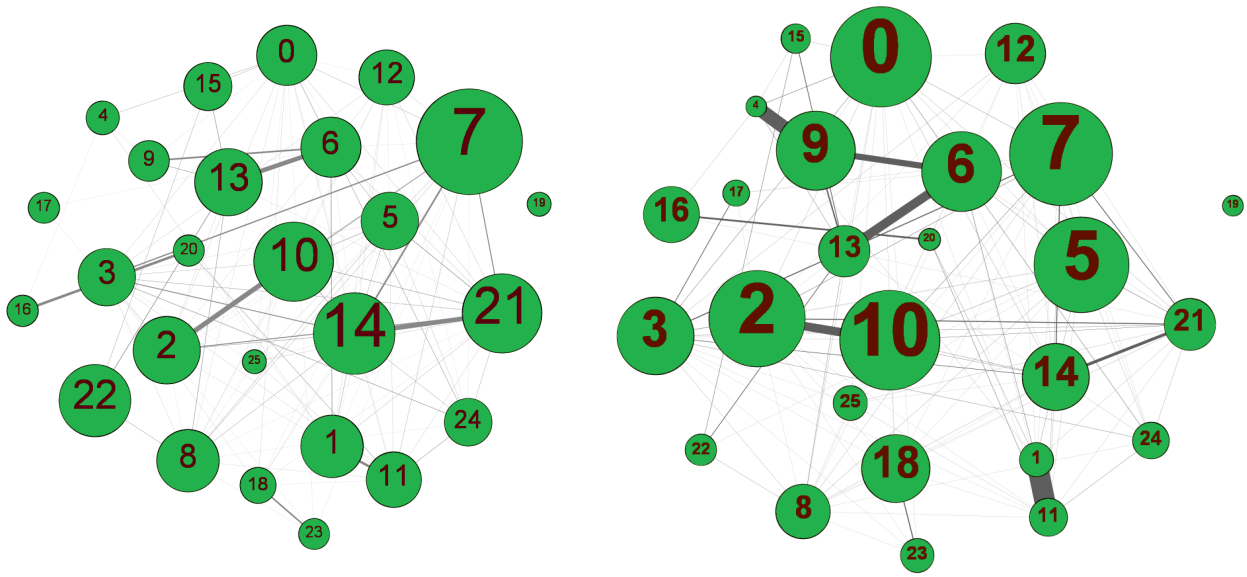
5.2 Clustering Techniques

Detecting the social structure of humans using clustering can aid the efficient transfer of messages in areas where the Internet infrastructure is not available [34]. The social graphs of WiFi datasets with RSSI greater than -60 and -70 dBm were visualized with a social network analysis tool called Gephi.[1] The visualization is based on PYDTNSIM results and the analysis of the datasets. The analysis of these social graphs can assist in a better understanding of how the clustering techniques partition the participantes and also how the policies of each PSNs routing algorithms behave. Balancing the number of nodes in clusters is an important feature of clustering algorithms to use both inter-cluster and intra-cluster message forwarding decisions.

Figures 5.3 to 5.10 demonstrate the graph visualization for different unclustered and cluster-based routing algorithms of PSNs in an unlimited resource scenario with a uniform message generation rate and random selection of source and destination pairs. In these visualizations, participants' devices are represented as nodes. The existence of contact between two participants is represented by an edge and edge thickness is proportional to the number of contacts between two nodes. The node size is proportional to the total number of packets forwarded by that node. This changes between the visualization because the policies of message transmission in each routing algorithm are different. Nodes with the same colour are considered to be located in the same cluster. The edge colours encode whether the contacts are among nodes in the same cluster (Green) or they belong to two different clusters (Pink). For unclustered algorithms, the colour of the edges is Gray.

Figure 5.3 demonstrates the social graphs of the Epidemic algorithm in the WiFi environment SHED5 with RSSI greater than -60 and -70 dBm. The epidemic is a multi-copy routing algorithm that works based on flooding. Because Epidemic is not clustered, all nodes are in the same colour in Figure 5.3. The weight of links is increased by changing transmission range from RSSI -60 dBm to -70 dBm, which means more contacts have been extracted by changing RSSI to -70 dBm. The size of the nodes also has been increased as a result of the transmission ranges of devices. The larger size of nodes indicates a higher number of message transmissions, which also shows energy consumption.

[1] https://gephi.org/



(a) WiFi -60 dBm

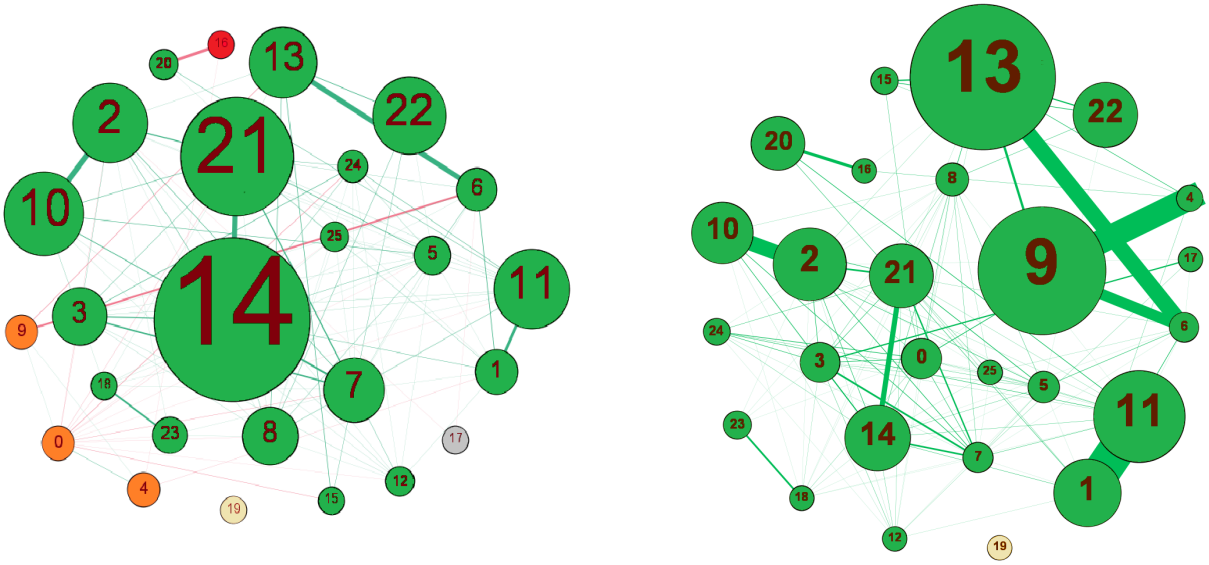
(b) WiFi -70 dBm

Figure 5.3: Epidemic social graphs - SHED5

Figure 5.4 and 5.5 demonstrate the social graphs of the Bubble-KClique and HCBF-KClique algorithms. KClique clustering is defined as the maximal union of all cliques of size k that can be reached through adjacent k-cliques, meaning that only fully connected subgraphs are found as clusters [52]. In KClique clustering, the existence of contact is more important than the strength of the contact and KCliques tries to maximize nodes' contact diversity in each cluster, rather than nodes' popularity.

If all nodes in a graph are the same colour, then the clustering algorithm has only detected one cluster, such as in Figures 5.4(b) and 5.5(b) (with the exception for node 19, which is disconnected from the other nodes). This implies that the entire graph is connected. KClique is not appropriate for a dense graph, because the number of clusters would be small. Having one cluster makes inter-clustering decisions useless and there would be no benefits of using different forwarding decisions as only one decision, intra-clustering would be applied to all messages. For example, in Bubble, only nodes' local popularity would be used as a factor of message forwarding and the global popularity factor would be ignored. Figure 5.4(a) visualized the KClique clustering techniques along with HCBF and Bubble routing algorithms for WiFi -60 dBm and -70 dBm. The number of clusters in WiFi -70 dBm is less than the WiFi -60 dBm. The fewer number of contacts causes fewer cliques, which leads to more clusters.

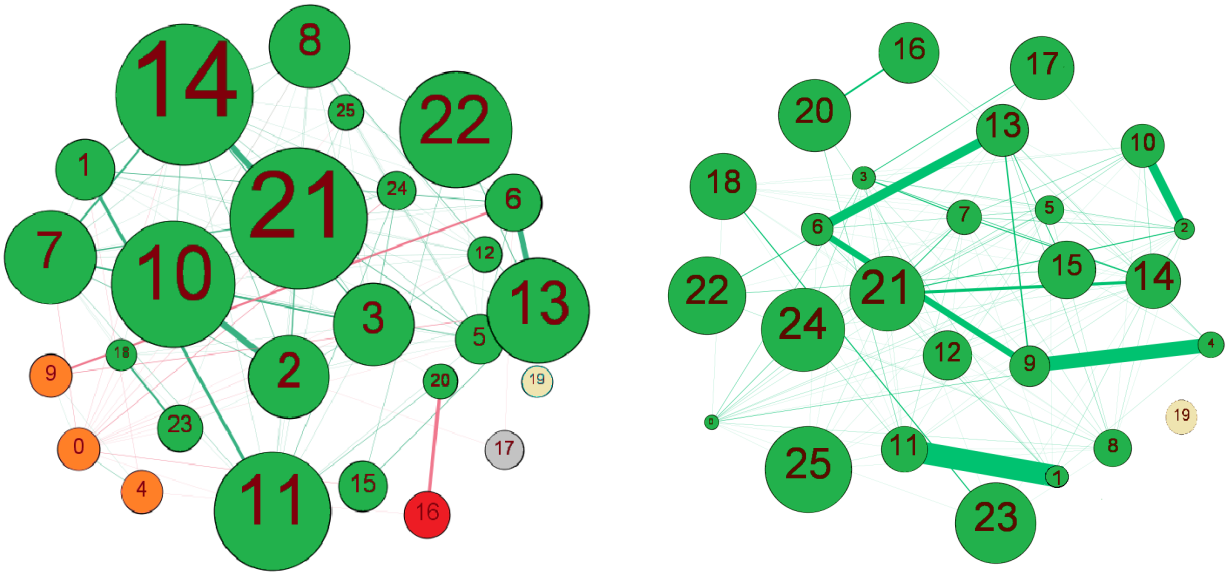
The previous figures consider contacts extracted from all durations of data recording in SHED5. These graph visualizations considered all contacts over the simulation time. However, in PYDTNSIM experiments, clustering is renewed after a specified period of time, usually one week [55]. Reclustering is important for following the current contact patterns of participants.



(a) WiFi -60 dBm

(b) WiFi -70 dBm

Figure 5.4: Bubble-KClique social graphs - SHED5



(a) WiFi -60 dBm

(b) WiFi -70 dBm

Figure 5.5: HCBF-KClique social graphs - SHED5

Figure 5.6 represents the social graphs of the first week of SHED5 experiments. KClique clustering detected 8 clusters for this graph. KClique allows overlap between clusters. Node 13 is located in two clusters. In this situation, if node 13 has contact with nodes in both overlapped clusters, such as node 22 or

9, intra-clustering decisions would be applied for forwarding decisions. KClique does not attempt to balance the clusters, leading one cluster to contain only one node and another having more than 15 nodes.

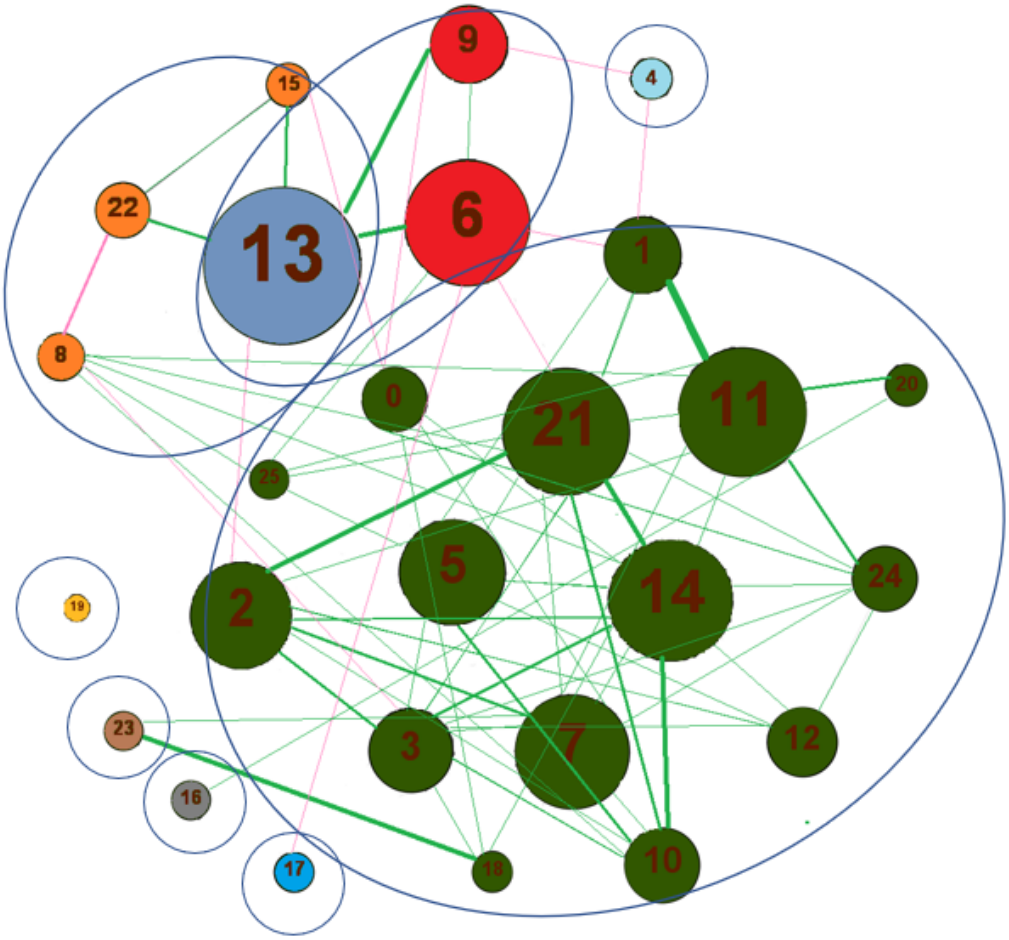
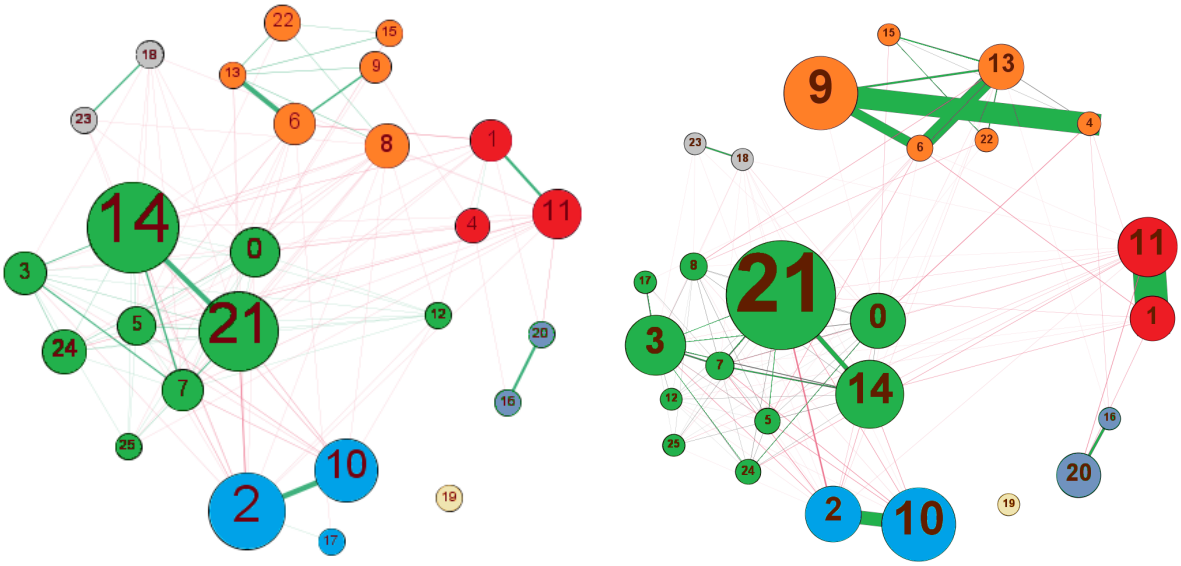


Figure 5.6: Social graphs of KClique clustering for the first week of SHED5 / WiFi with -70 dBm

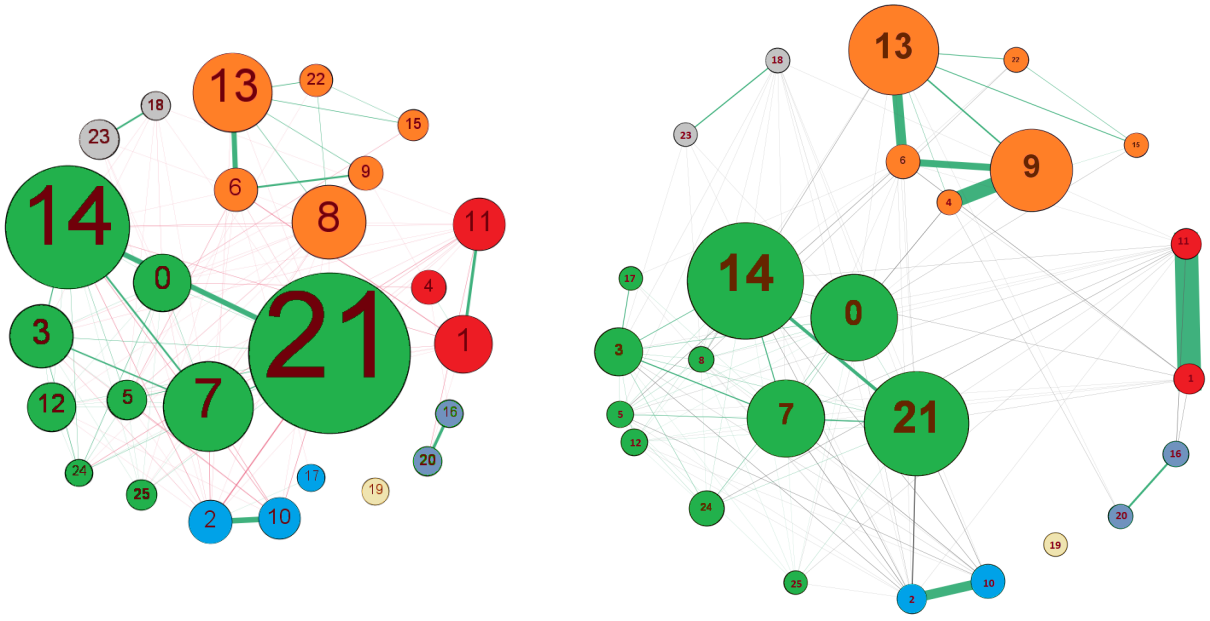
Figures 5.7 and 5.8 visualized the social graphs of Bubble and HCBF using the Louvain clustering algorithm. The Louvain algorithm determines the number of clusters dynamically and has detected 6 clusters for different transmission ranges of WiFi datasets. Changing the RSSI value from -60 dBm to -70 dBm does not change the number of the clusters. A few nodes nodes changed clusters with the modification of the transmission range, but most nodes are still in their previous clusters and the topology of the social graph clustering is fairly stable. Louvain does not attempt to balance the clusters leading one cluster containing two nodes and another having more than 10 nodes. As Louvain tries to increase the edges weight in each cluster and minimize it between clusters.



(a) WiFi -60 dBm

(b) WiFi -70 dBm

Figure 5.7: Bubble-Louvain social graphs - SHED5



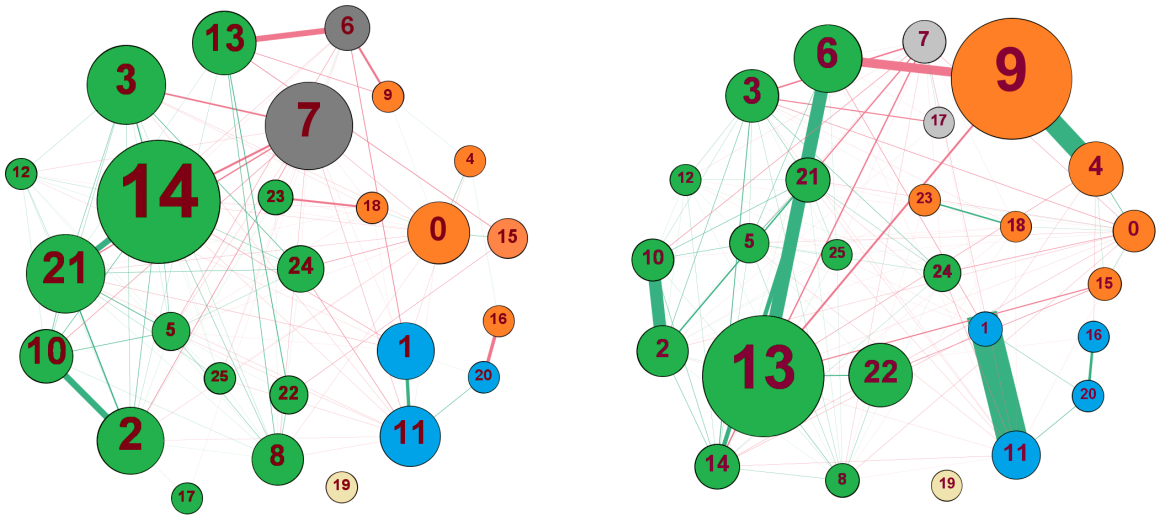
(a) WiFi -60 dBm

(b) WiFi -70 dBm

Figure 5.8: HCBF-Louvain social graphs - SHED5

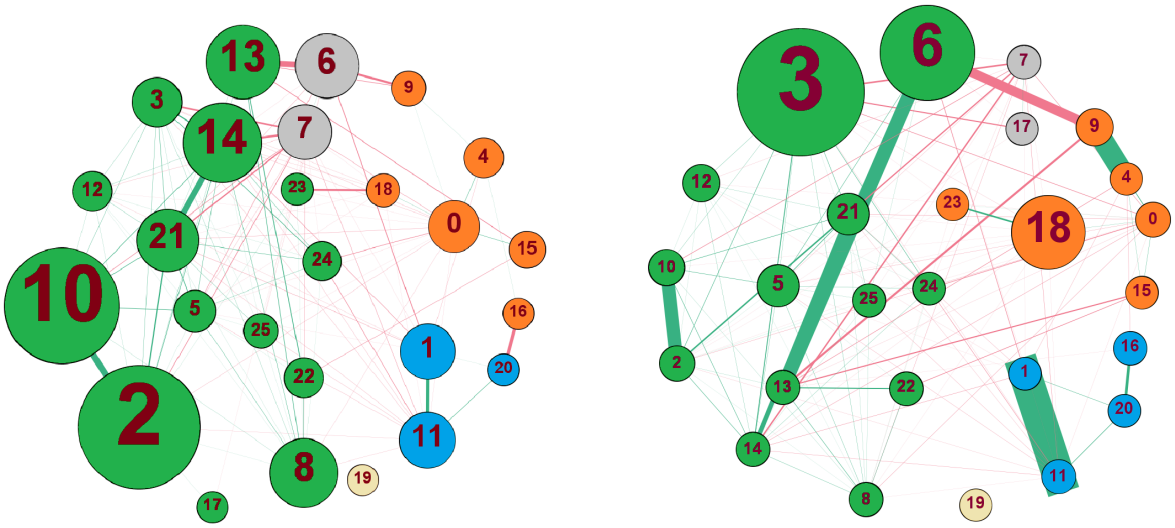
Figures 5.9 and 5.10 represent the HCBF and Bubble algorithms that use AGKmeans as their clustering

algorithms. The number of clusters in AGKmeans clustering is based on a pre-defined number (k=4 in these graphs). For nodes that have no contacts, AGKmeans creates another cluster, leading to node 19 landing in a fifth cluster. In each AGKmeans social graph, five different colours are visible.



(a) WiFi -60 dBm **(b)** WiFi -70 dBm

Figure 5.9: Bubble-AGKmeans social graphs - SHED5



(a) WiFi -60 dBm **(b)** WiFi -70 dBm

Figure 5.10: HCBF-AGKmeans social graphs - SHED5

By changing the transmission range in each environment, several nodes moved to other clusters, but the topology of the social graphs remained unchanged, providing confidence that AGKmeans is a relatively stable

algorithm. AGKmeans is not a balanced clustering algorithm, as one cluster contains only 2 nodes other has more than 10 nodes. AGKmeans tries to keep the nodes with high dependency and connection in the same cluster but is limited by the predefined k. AGKmeans uses the diversity of nodes as a factor of initial centroid selection tries to select nodes with high local diversity of contacts in the same cluster.

Different clustering algorithms (KClique, Louvain, and AGKmeans) have different performance in node balancing, stability, the parameters clustered against and the degree of overlap allowed between clusters. Each of these capabilities can affect routing algorithms policies and the performance of PSNs routing algorithms. The KClique clustering approach tries to find full subgraphs as clusters and increase the number of contacts in each cluster. Louvain tries to increase the edge weight in each iteration and keep nodes with strong contacts in the same cluster. AGKmeans also tries to increase the edges weight in each cluster, but the initial centroid selection considers both the number and the strength of contacts. None of the implemented clustering algorithms balance the number of nodes in each cluster. KClique is the worst in balancing as it only found the full subgraphs, which can increase the number of clusters with one node. The social graphs visualized how the transmission ranges can increase the opportunity of catching more contacts by reducing the signal strength of devices required to established a contact from -60 dBm to -70 dBm. PSNs routing algorithms, Epidemic, HCBF, and Bubble different message forwarding policies can lead to different message transmission which is shown in social graphs by node sizes.

5.3 Usecase Scenarios Results

Comparing the clustering and routing algorithms of PSNs needs to be done in the context of a use case. Three use case scenarios have been implemented in this thesis: a) random source and destination, b) plant and habitat monitoring, and c) disaster relief. These use case scenarios cover a range of possible selection of source and destination pairs of messages from selecting source and destination from the same dynamic cluster to selecting source and destination from different dynamic and static clusters. Delivery ratio, delivery delay, energy consumption, and execution time have been computed for each of these use cases. The scenarios were run under both infinite and limited resources. Results were generating using the PYDTNSIM simulator.

5.3.1 Infinite Resource for Random Source and Destination Scenario

In the infinite resource case, the bandwidth and buffer space of nodes have no limitations. No expiry time for packets is specified and the size of packets is considered zero. In this scenario, the message generator generates one message in each duty cycle. The source and the destination of the nodes are selected randomly and the reclustering epoch is set to one week. With infinite resources, each device can transfer all the messages that meet the policies of the routing algorithm for forwarding to other devices in their contact. Each node can keep as many messages in its buffer as it receives without needing to remove other messages. Twenty-five runs of the PYDTNSIM simulator were performed for each algorithm with different random seeds.

The average delivery ratios for SHED1 and SHED5, for 7 different algorithms (Epidemic, BubbleAGKmeans, HCBFAGKmeans, BubbleLouvain, HCBFLouvain, BubbleKClique, and HCBFKClique) in two environments (GPS and WiFi), with three different transmission ranges are represented in Figures 5.11 and 5.12. Increasing the transmission range in each environment increases the delivery ratio as a result of the increasing contact opportunity. As is shown in Table 4.6, the number of contacts increases when increasing the transmission ranges in both environments. The dependency of delivery ratio and the number of contacts is also apparent in the comparison of WiFi and GPS in SHED1 and SHED5. In SHED1, GPS has a lower delivery ratio than WiFi and SHED5 GPS has a higher delivery ratio than WiFi. This is the result of having more chances for meeting nodes with higher social factors to carry the messages towards their respective destinations.

Tables 5.1 and 5.2 demonstrate the standard deviation for the delivery ratio in the GPS and WiFi environments of SHED5. As the delivery ratios are the means of 25 of different runs of experiments, the standard deviation can measure the dispersion of each run. All of the values both in GPS and WiFi are very small, indicating consistency between runs. The remainder of the experimental runs had similar and small variations as shown in tables 5.1 and 5.2.

Table 5.1: Standard Deviation of Delivery ratios for SHED1 GPS for 25 Different Runs

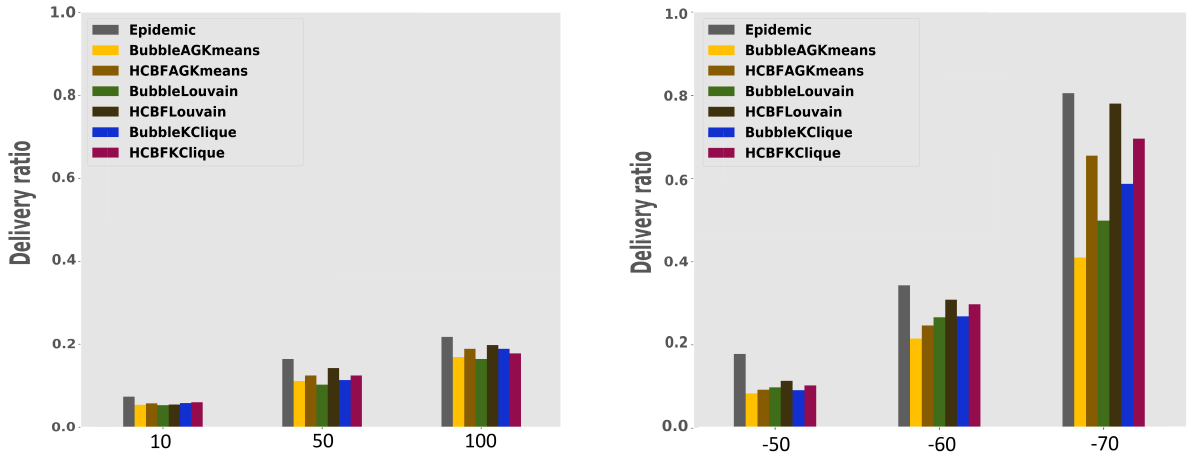
	10 m	**50 m**	**100 m**
Epidemic	0.0028	0.0058	0.0085
BubbleAGKmeans	0.003	0.0057	0.0088
HCBFAGKmeans	0.0018	0.0081	0.0085
BubbleLouvain	0.0021	0.0059	0.0097
HCBFLouvain	0.0057	0.0041	0.0055
BubbleKClique	0.0048	0.0052	0.0060
HCBFKClique	0.0033	0.0057	0.0095

Table 5.2: Standard Deviation of Delivery ratios for SHED1 WiFi for 25 Different Runs

	-50 dBm	**-60 dBm**	**-70 dBm**
Epidemic	0.0028	0.0061	0.0074
BubbleAGKmeans	0.0041	0.0066	0.0073
HCBFAGKmeans	0.0016	0.0061	0.0079
BubbleLouvain	0.0033	0.0059	0.0067
HCBFLouvain	0.0047	0.0046	0.0056
BubbleKClique	0.0048	0.0040	0.0072
HCBFKClique	0.0012	0.0045	0.0084

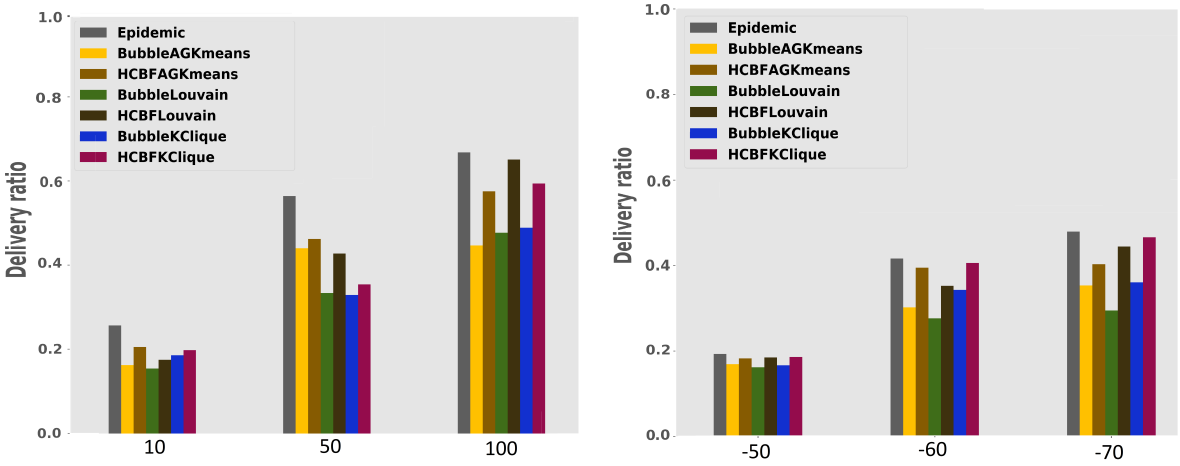
Epidemic routing has the highest delivery ratio among all algorithms as expected. The Epidemic algorithm is always optimal for the delivery ratio under infinite resource conditions as it will forward messages to all nodes in contact which do not already contain them. Generating multiple copies of the same message, and routing them separately, maximized the message delivery ratio by searching all possible paths.

Among cluster-based routing algorithms, the algorithms that use HCBF have a higher delivery ratio in comparison to the algorithms that use Bubble. This shows that adding diversity and interaction between clusters by HCBF for forwarding decisions has a positive effect on the delivery ratio.



(a) GPS (b) WiFi

Figure 5.11: Delivery ratio SHED1 Random Src/Des - Unlimited Resources - Uniform Message Generation



(a) GPS (b) WiFi

Figure 5.12: Delivery ratio SHED5 Random Src/Des - Unlimited Resources - Uniform Message Generation

By analyzing the number of contacts in GPS and WiFi, it can be concluded that the delivery ratio in WiFi environment is higher than GPS for these datasets. For the same or even fewer contacts in WiFi, a higher delivery ratio can be reached in comparison to GPS. For example, in SHED5, WiFi with RSSI greater than -60 dBm has 1144 contacts and GPS with transmission range less than 10 metres have 1912 contacts. The number of contacts in GPS is less than WiFi, and Figure 5.12 demonstrates that the delivery ratio of all algorithms in WiFi with RSSI greater than -60 dBm is higher than GPS 10 metres. The reason for higher delivery ratio in WiFi environment can be explained by Figures 5.13 and 5.14, which demonstrate the participation of nodes in contact of GPS 10 metres and WiFi -60 dBm of SHED5.

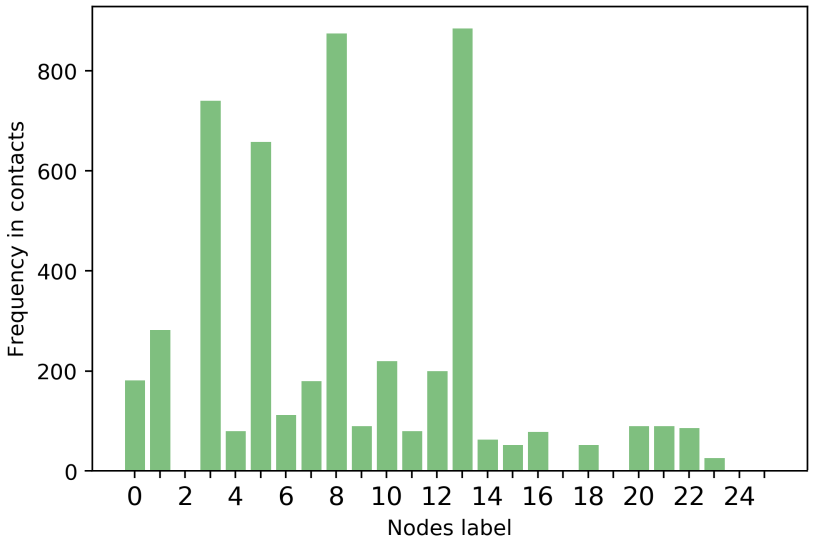


Figure 5.13: Node participation in contacts / GPS 10 metres / SHED5

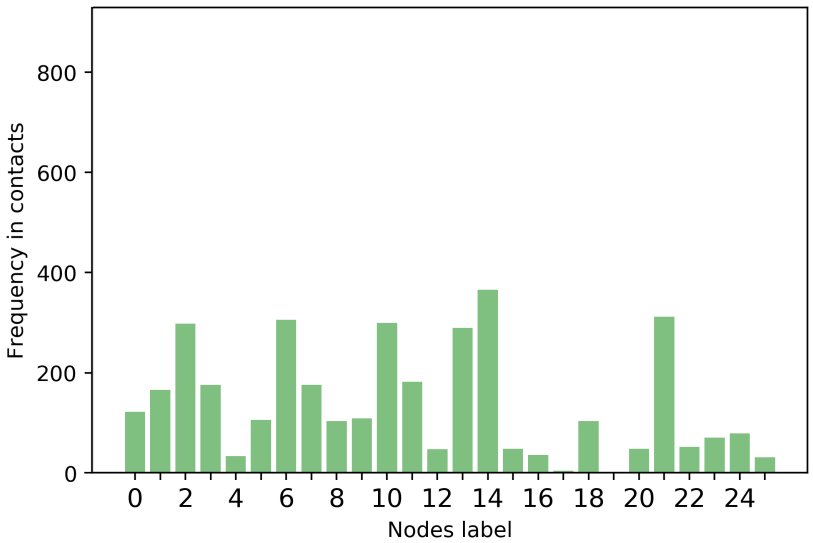


Figure 5.14: Node participation in contacts / WiFi -60 dBm / SHED5

Based on Figure 5.13, nodes 2, 17, 19, 24, and 25 have no contacts with other nodes in GPS 10 metres. However, in WiFi -60 dBm, only node 19 has no contacts with other nodes. Figures 5.13 and 5.14 indicate that nodes' participation in WiFi is more balanced than GPS. Lack of participation of 5 nodes in contacts of GPS 10 metres causes approximately 20% reduction in delivery ratio due to the impossibility of delivering packets when these nodes are randomly chosen as either source or destination.

There are other examples where WiFi acts better than GPS in situations that the number of contacts is approximately the same. In WiFi -70 dBm with 7321 contacts and GPS 50 metres with 8362 of SHED5, the delivery ratio of WiFi -70 dBm is higher than GPS. Furthermore, in WiFi with RSSI greater than -60 dBm of SHED1, the number of contacts is 821 and in GPS with transmission range less than 10 metres the number of contacts is 901. However, the delivery ratio of the WiFi environment in this situation is higher than GPS. In all these scenarios, GPS has a higher number of nodes that have not participated in message transmission and cause a reduction in delivery ratio.

The reason for the unbalanced participation of nodes in GPS contacts is related to the fact that the participants of SHED1 and SHED5 are students and staff at the university. Participants spend their time in indoors offices and classrooms lower the number of GPS records. Figures 5.15 and 5.16 which demonstrate the contacts diversity of each nodes justifies this fact.

To analyze the effect of contact frequency on delivery ratio, consider WiFi in SHED1 and SHED5 environments. The WiFi with RSSI greater than -50 dBm has 821 contacts in SHED1 and 360 in SHED5 and WiFi with RSSI greater than -60 dBm have 3771 contacts in SHED1 and 1144 in SHED5. However, in both cases, the delivery ratio in SHED5 is higher than SHED1. Based on section 5.1, which demonstrates the frequency of contacts duration in each meeting, SHED1 contacts are longer-lasting than SHED5. In SHED1, several contacts last for 100 duty cycles. However, in SHED5 the maximum contact duration is 16 duty cycles. For GPS with transmission range less than 50 metres in SHED1 have 2524 contacts and 10 metres in SHED5 that have 1912 contacts. SHED5 has fewer contacts, but it has a higher delivery ratio which is the effect of sporadic contacts. By having the same number of contacts, sporadic contacts can meet a variety of nodes and this increases the opportunity of finding appropriate carrier nodes and increasing the delivery ratio. As Figures 5.15 and 5.16 indicate, with the same number of contacts, WiFi has a higher number nodes with diverse contacts. The delivery ratio of SHED1 and SHED5 demonstrated that increasing the diversity of contacts can increase the delivery ratio.

Figures 5.17 and Figure 5.18 demonstrate the average energy consumption, calculated by counting the number of packet transmissions. For example, based on Figure 5.17, the energy consumption of Bubble-AGKmeans in the GPS environment with the transmission range of 50 metres is 2. It means most of the packets reached their destinations within two hops.

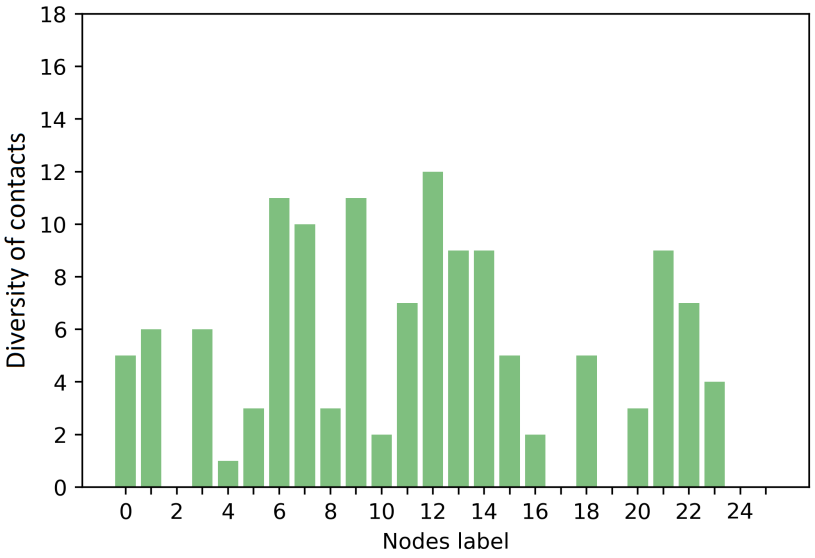


Figure 5.15: Node contact diversity / GPS 10 metres / SHED5

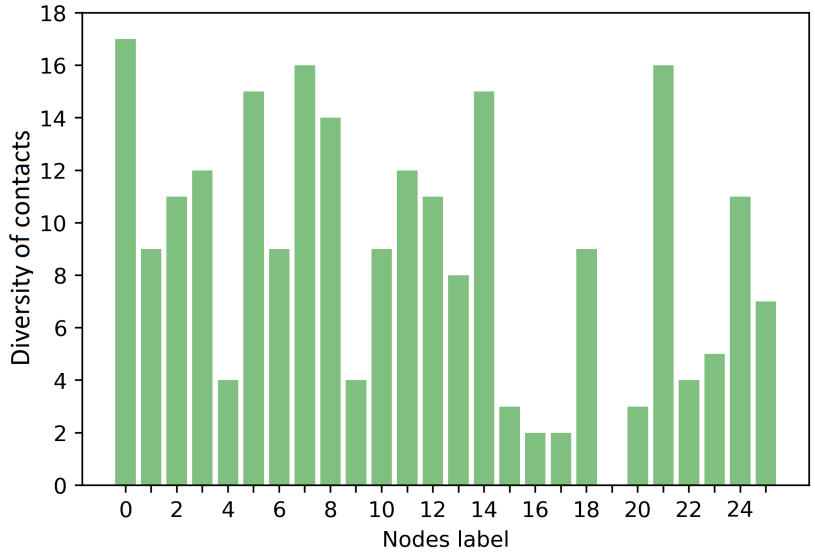
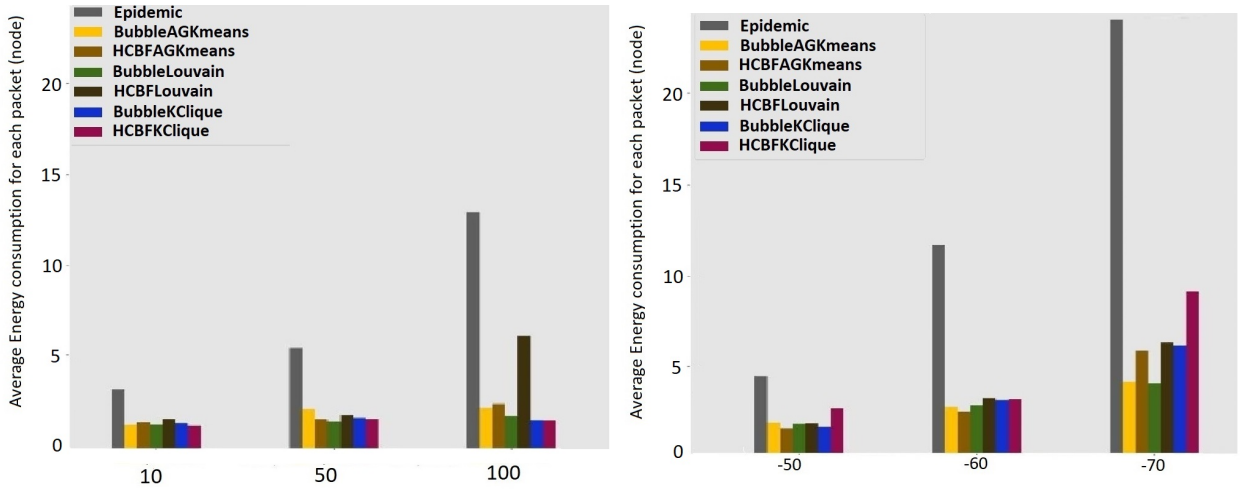


Figure 5.16: Nodes contact diversity / WiFi -60 dBm / SHED5

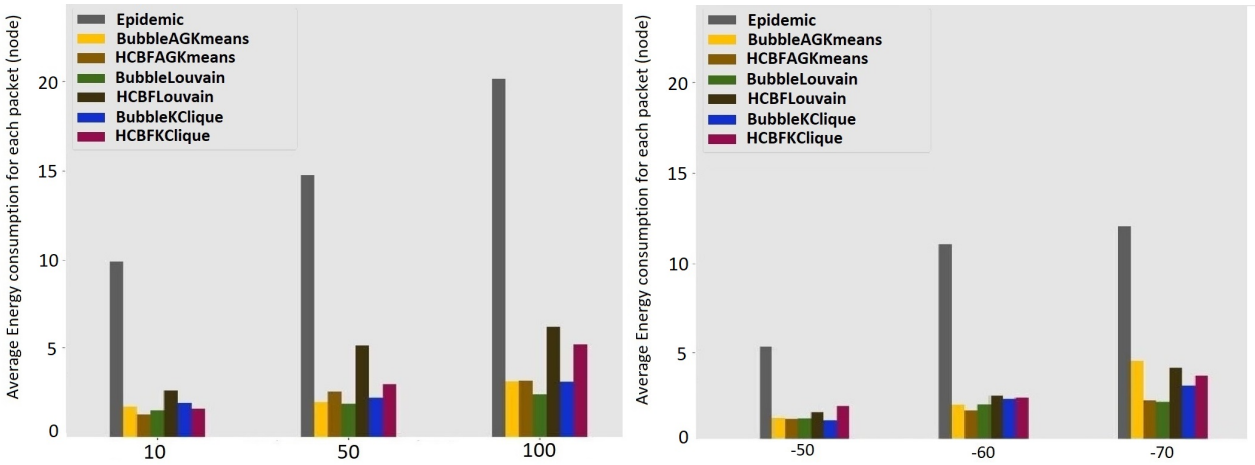
In most cases, energy consumption depends on the delivery ratio. As expected, Epidemic routing has the worst energy consumption as is shown in Figure 5.17. The average energy consumption for packets in the Epidemic routing algorithm of WiFi with the transmission range of RSSI greater than -70 dBm is 24, meaning more than half of the nodes in the network have transmitted the packet, as the number of participants in this dataset is 39.



(a) GPS

(b) WiFi

Figure 5.17: Average Energy Consumption SHED1 Random Src/Des - Unlimited Resources - Uniform Message Generation



(a) GPS

(b) WiFi

Figure 5.18: Average Energy Consumption SHED5 Random Src/Des - Unlimited Resources - Uniform Message Generation

Among single-copy routing algorithms, in most cases, the algorithms that use HCBF as their routing algorithms consume slightly more energy than the algorithms that use Bubble. HCBF-Louvain consumes more energy than the other HCBF algorithms. Higher energy consumption in the HCBF routing algorithm is due to messages that are delivered only by the HCBF algorithm [55]. As the delivery ratio of HCBF algorithms is higher than Bubble, the packets that are delivered by HCBF, but not by Bubble, usually need to be carried by more intermediate nodes and this leads to more transmissions. These packets can affect the

total energy consumption of HCBF algorithms.

Figure 5.19 demonstrates the energy consumption for 20 messages which have been delivered in all 7 different algorithms in unlimited resources scenario. Based on this figure, HCBF algorithms consume less energy than Bubble algorithms for the same set of messages. Furthermore, the ANOVA test shows that HCBF-AGKmeans and HCBF-Louvain have the best energy consumption, respectively. HCBF-KClique energy consumption has a higher variance than HCBF-AGKmeans and HCBF-Louvain. Epidemic energy consumption has the highest variance and has the worst overall performance.

In PSNs, eventual message delivery is enough, but several PSNs use cases benefit from lower delivery delay such as in disaster relief scenarios where one message can save people. In these figures, the maximum and minimum, median and quartiles of delay for each algorithm are represented. The top and the bottom of the box represent the first and the third quartile of the delay, whereas the line in the box represents the median delay. The outliers are messages which their delay have a noticeable difference from the rest of messages delay.[2] The execution time of the algorithms is affected by the high number of messages that require to be forwarded. HCBF forwarding decisions need more calculations than Bubble which increased the execution time of HCBF algorithms and in clustering algorithms, the number of iteration for forming clusters can affect the execution time of the algorithms.

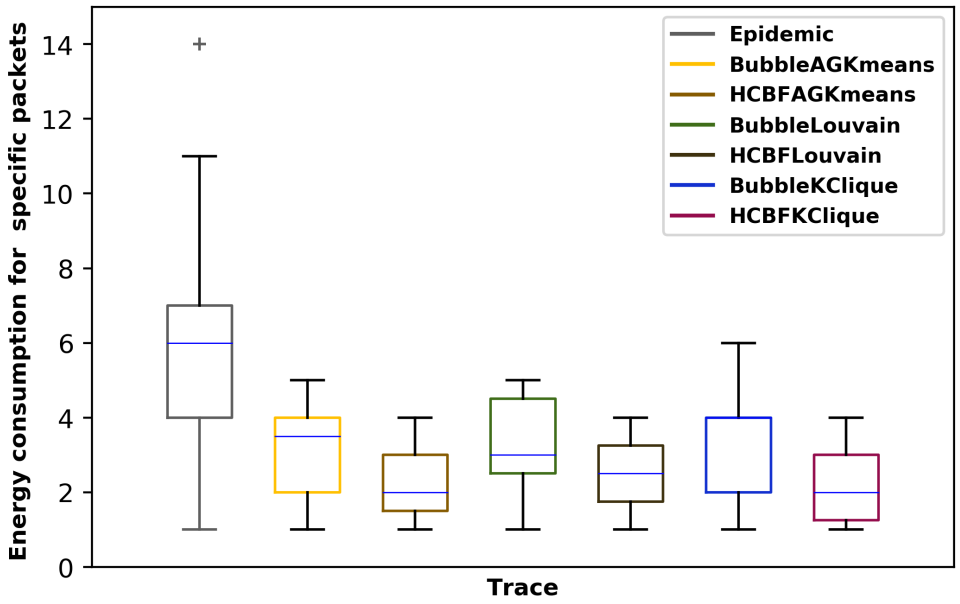
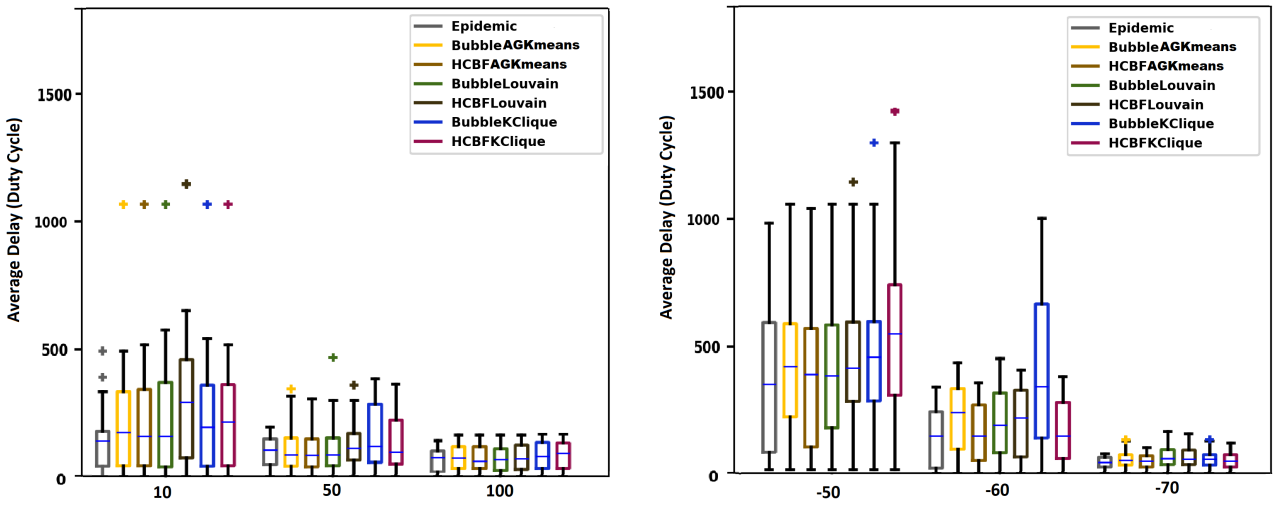


Figure 5.19: Energy consumption for 20 specific packets SHED5 GPS 10 metres

The delivery delay is highly dependent on the transmission range. By increasing the transmission ranges, the delay has been decreased noticeably. For example, for the messages which have the highest delivery delay in the transmission range of -50 dBm, the delivery delay has a minimum improvement of 65%.

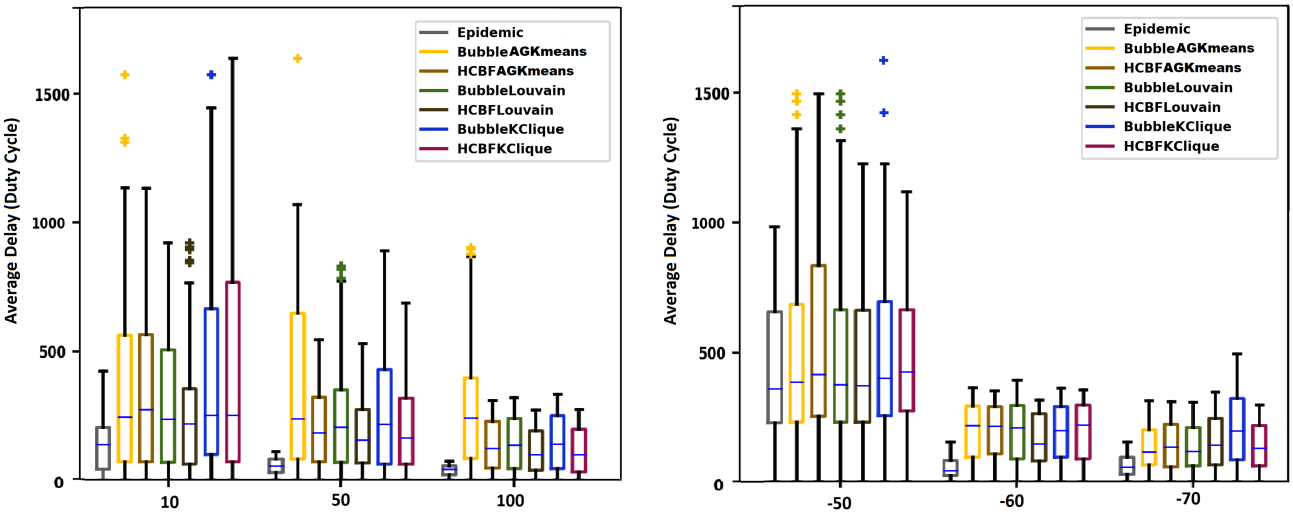
[2] https://towardsdatascience.com/understanding-boxplots-5e2df7bcbd51

The Epidemic algorithm has the best delivery delay among all algorithms in all transmission ranges as expected. In most scenarios, the delivery delay of the algorithms using HCBF is lower than the algorithms using Bubble which may be caused by HCBF selecting appropriate intermediate nodes. Although the comparison among the delay affected by the clustering algorithm is difficult because of the sensitivity of delay by variant messages delivered in each algorithm, HCBFLouvain has a lower delay, especially in SHED5.



(a) GPS (b) WiFi

Figure 5.20: Delivery Delay SHED1 Unlimited Resources Random Src/Des



(a) GPS (b) WiFi

Figure 5.21: Delivery Delay SHED5 Unlimited Resources Random Src/Des

5.3.2 Resource Limited Scenario for Random Source and Destination

Infinite buffer space for devices and unlimited TTL and zero packet size for packets are unrealistic scenarios that would not happen in real networks. Unlimited resource scenarios results can be used as a baseline for comparison of algorithms. For the case of the limited resources, nodes buffer space was constrained. Generated packets expire after a finite time, and each packet has a finite payload size.

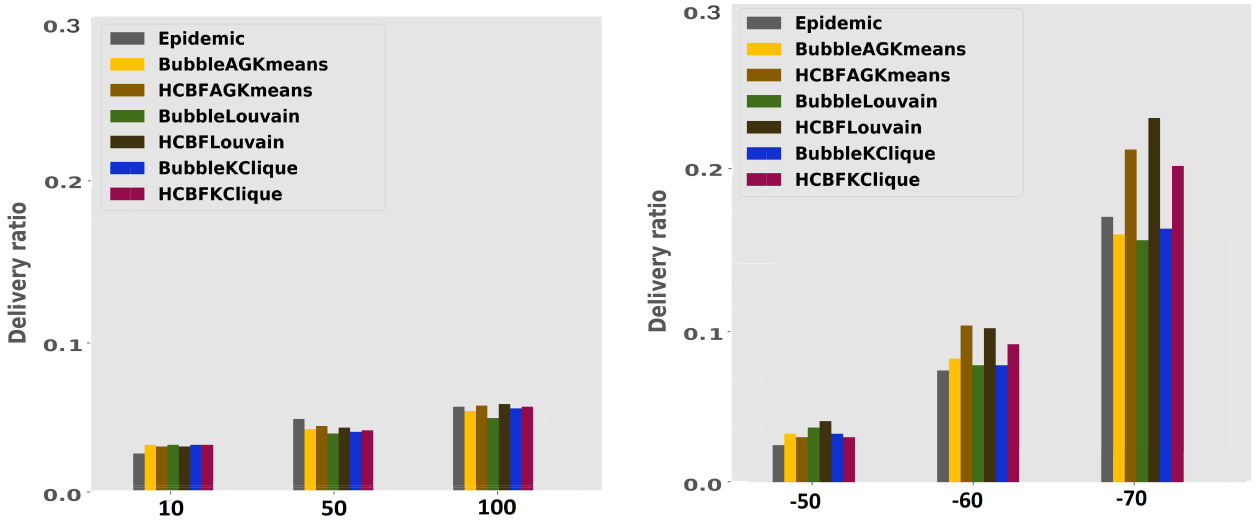
Figure 5.22 and Figure 5.23 represent a complete comparison of the delivery ratio for random source and destination with constarinted resources in GPS and WiFi environments of SHED1 and SHED5 . The delivery ratio for the 7 different algorithms, in GPS and WiFi environments, with three different transmission ranges for each environment, has been run 25 times. The variation among these run is approximately the same as unlimited resources scenario.

By increasing the transmission range in each environment, the delivery ratio is also increased due to the higher number of contacts captured. The Epidemic routing delivery ratio has a noticeable reduction compared to the infinite resources scenario. Algorithms that use HCBF have a higher delivery ratio in comparison to Bubble. Among the single-copy routing algorithms, HCBF-Louvain and HCBF-AGKmeans have the best delivery ratio.

By constraining the resources, datasets with sporadic contacts have a higher delivery ratio. By comparing SHED1 and SHED5 for cases with approximately the same number of contacts, the delivery ratio of SHED5 is higher than SHED1 which is the effect of contact stability as discussed in random source and destination with unlimited network resources scenario.

By constraining the resources, the WiFi dataset has a higher delivery ratio than GPS the same as the unlimited resources scenario. For example, for GPS 10 metres which have 901 contacts and for WiFi with RSSI greater than -50 dBm which have 821 contacts in SHED1 based on Table 4.6, the delivery ratio of algorithms in WiFi environments is higher than GPS. This is the effect of the balanced participation of nodes in WiFi contacts and having more unique contacts.

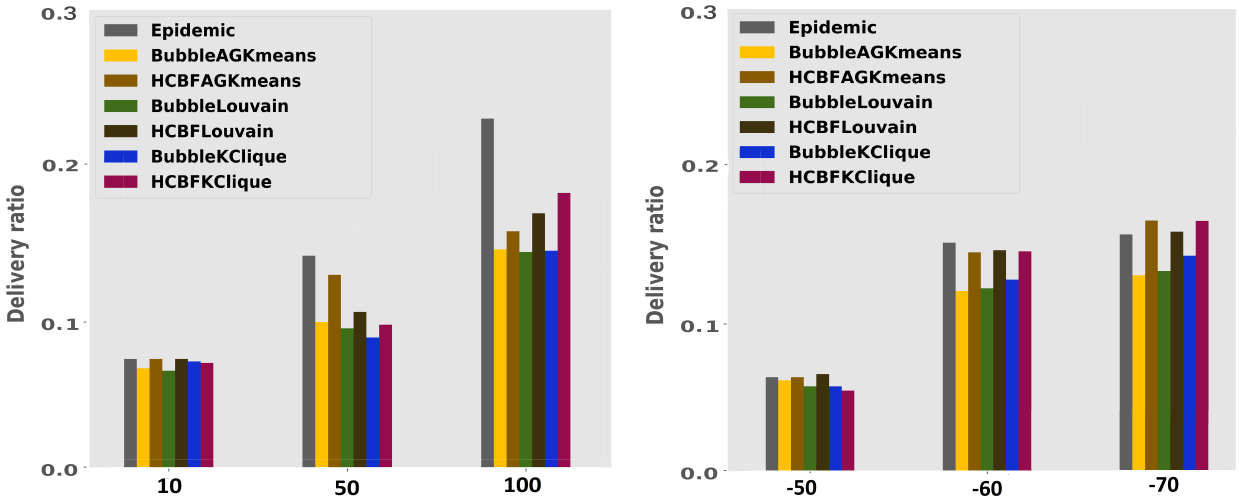
Reduction in the delivery ratio for Epidemic routing after applying resource limitation can be explained by Figure 5.24, which demonstrates the frequency of packet copies generated for each packet. Based on this figure, most of the packets in the Epidemic routing algorithm have only one copy in the network, but some packets have been copied for more than 20 times. By having 29 participants in this dataset, it means in this case, most of the nodes in the network have forwarded these packets, consuming energy and buffer space. In single-copy algorithms, each message only can have one copy in the network allowing single-copy algorithms to consume fewer resources than multi-copy algorithms. In limited-resource scenarios, the Epidemic algorithms performance decreases due to the effect of occupied buffer space and extra bandwidth consumed by a high number of copied packets.



(a) GPS

(b) WiFi

Figure 5.22: Delivery ratio SHED1 Random Src/Des Limited Resources - 7 days TTL - Uniform Message Generation



(a) GPS

(b) WiFi

Figure 5.23: Delivery ratio SHED5 Random Src/Des Limited Resources - 7 days TTL - Uniform Message Generation

Figures 5.25 and 5.26 demonstrate energy consumption which is calculated by message transmissions for limited resource scenarios. Single-copy algorithm energy consumption improves after resource limitation, Epidemic routing energy consumption improvement is noticeable. The reason for energy consumption improvement can be explained by the limited buffer space of nodes and TTL of packets that leads to removing

older packets which may pass many intermediate nodes. As Epidemic has multiple copies for packets, many old packets would be removed by limited buffer space which causes a reduction in message transmissions and energy consumption.

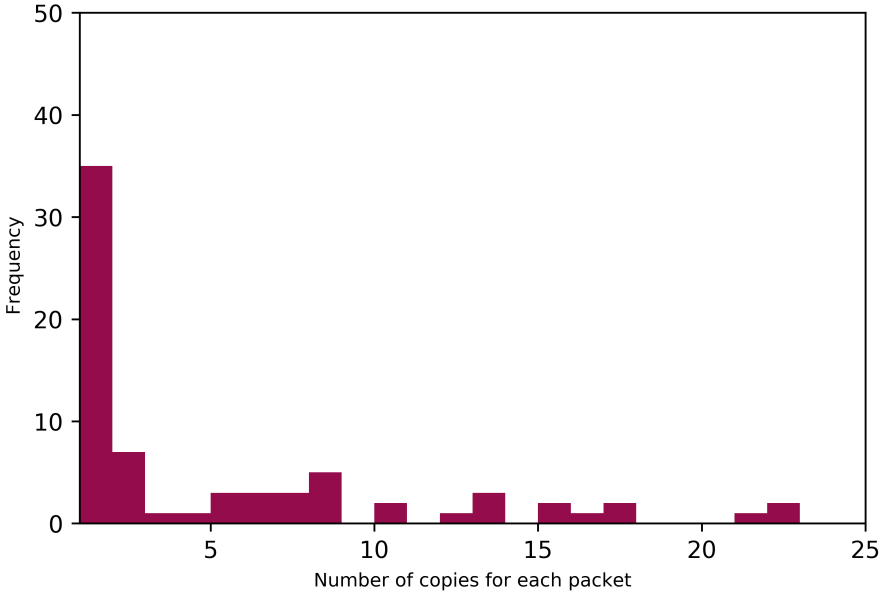
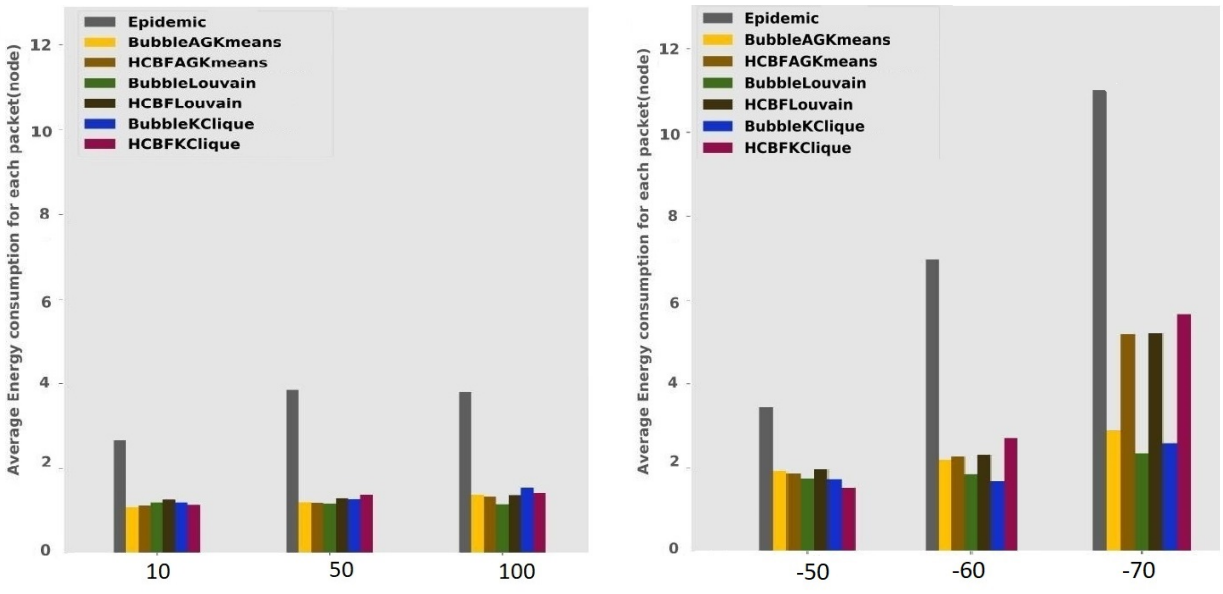
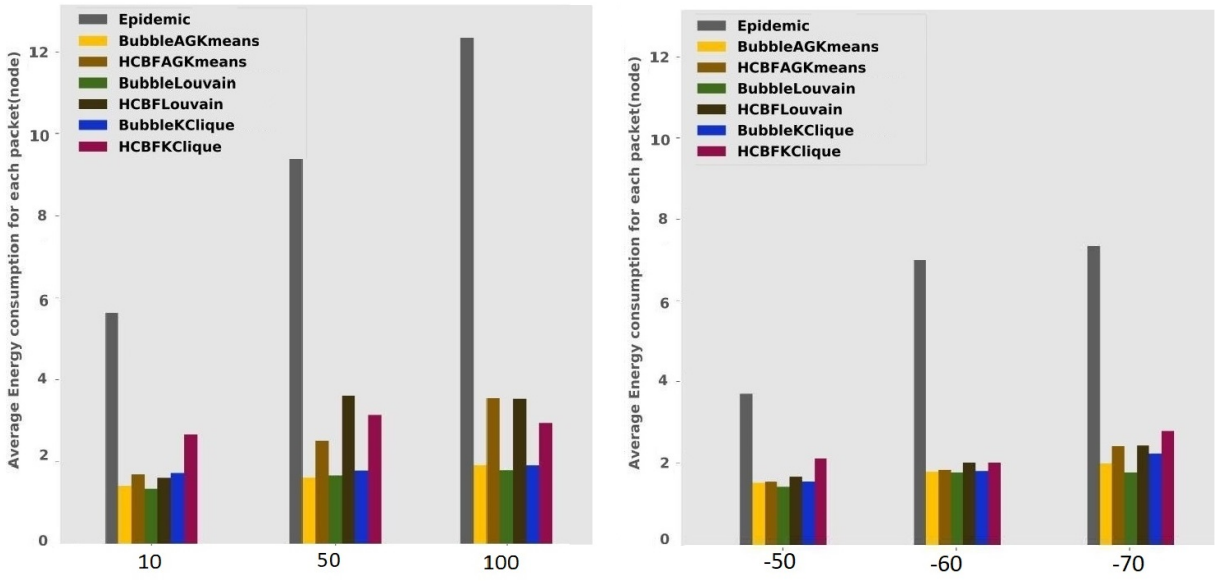


Figure 5.24: Frequency of the number of message copies in Epidemic routing - GPS 50 metres - SHED5



(a) GPS **(b)** WiFi

Figure 5.25: Average energy consumption SHED1 Random Src/Des - Limited Resources - 7 days TTL - Uniform Message Generation



(a) GPS

(b) WiFi

Figure 5.26: Average energy consumption SHED5 Random Src/Des - Limited Resources - 7 days TTL - Uniform Message Generation

5.3.3 Plant and Habitat Monitoring Scenario with Limited Resources

Plant and habitat monitoring scenarios have both static and dynamic devices. SHED1 and SHED5 are insufficient for these uses case scenarios as they only contain dynamic participants of university students and staff. The Termite emulator dataset has been used for the plant and habitat monitoring scenario. This dataset contains 30 participants, 9 static and the rest are considered to be dynamic. Static nodes are placed in a way that each pair of static nodes can achieve contact with two or fewer hops. Statics nodes are selected as the source and dynamic nodes are the destination of packets. In this scenario, packets would be expired after 7 days and each buffer can store 10 packets.

Figure 5.27 demonstrates the delivery ratio for the limited resources plant and habitat monitoring scenario with the Termite dataset. In this scenario, the one message is generated per duty cycle. The message size was set such that each node can carry 10 messages. The Epidemic algorithm has the best delivery ratio among all algorithms. Comparing the delivery ratio of this scenario with Random Source and Destination scenario which have the same message generation rate and resources capacity unveiled the fact that cluster-based routing algorithms cannot act efficiently for gaining a high delivery ratio in static datasets.

The reason for the reduction of delivery ratio in the cluster-based algorithm is the static part of datasets such that only specific nodes be selected as the carrier of messages. In datasets with static participants, popular nodes and nodes with diverse contacts are selected as the carrier node for most of the packet transmissions. Because the structure of static nodes is the same in simulation time.

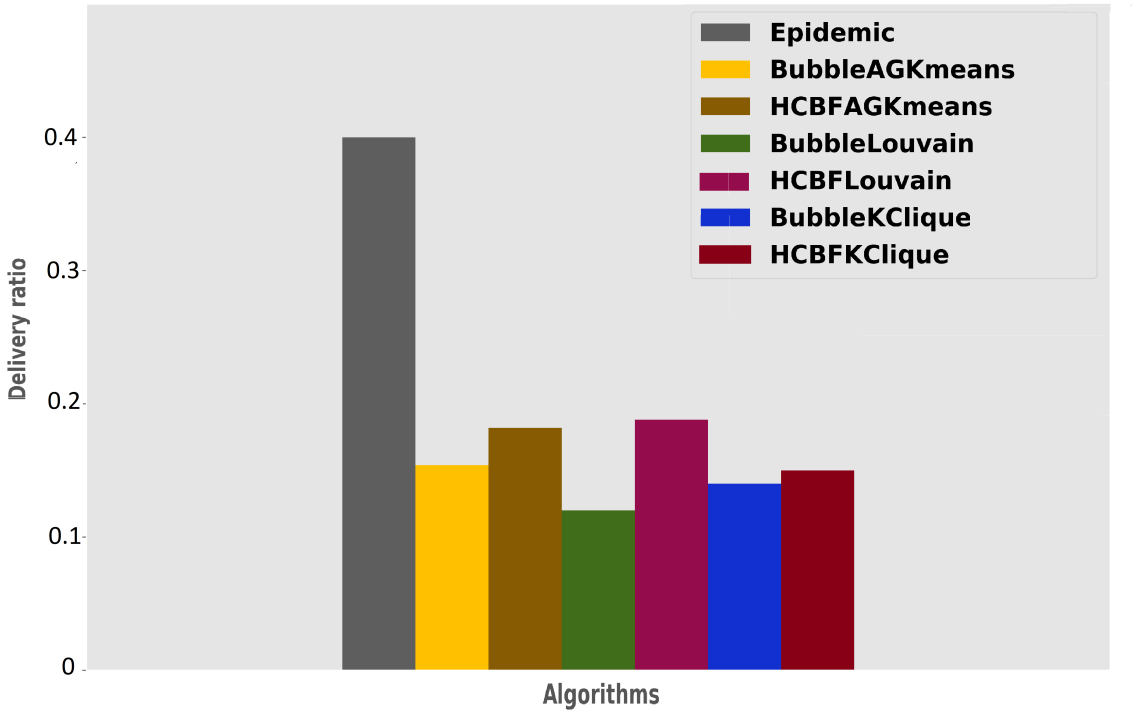


Figure 5.27: Delivery ratio - Limited resources plant and habitat monitoring - 7 days TTL - Uniform Message Generation - Termite

Figure 5.28 demonstrates the participation of Termite static nodes as carrier nodes. Based on this figure, in most cases, nodes 25, 26, and 28 are selected as carrier nodes. The buffer space occupied by a large number of messages in the selected carrier nodes causes a reduction in delivery ratio as these nodes would drop packets while they have no sufficient buffer for storing packets. In this case, the source or adjacent nodes of popular and diverse nodes may be better carriers with more free buffer spaces.

Figure 5.29 demonstrates the average energy consumption for delivering each packet in the limited resources plant and habitat monitoring scenario. The multi-copy routing algorithm, Epidemic, has the highest average energy consumption. Almost every message has been carried by more than half of the network nodes. The reason for the high energy consumption of the Epidemic algorithm is the static part of the dataset in which all nodes receive the packets at least once. In this scenario, all 9 static participants received the generated packets as these participants are vertexes of a connected graph.

The worst case of energy consumption in single-copy routing algorithms is 5, on average, which implies packets have been delivered to their destination by the average transmission of five hops. The energy consumption in the static dataset is low because of the unchanged social behaviour of nodes.

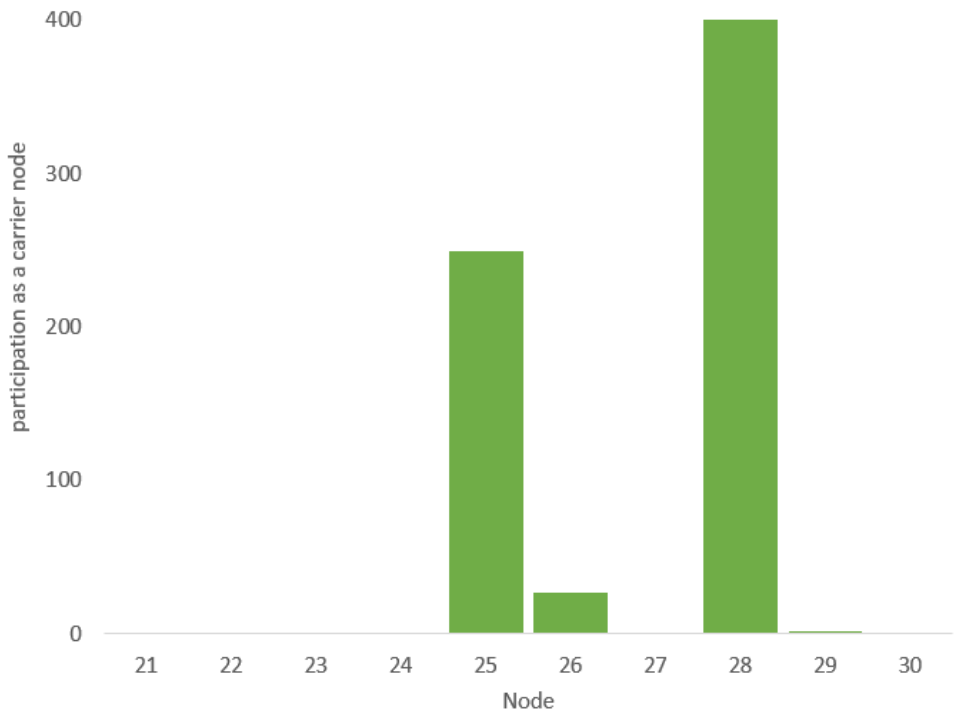


Figure 5.28: Static Nodes Participation as Carrier Nodes

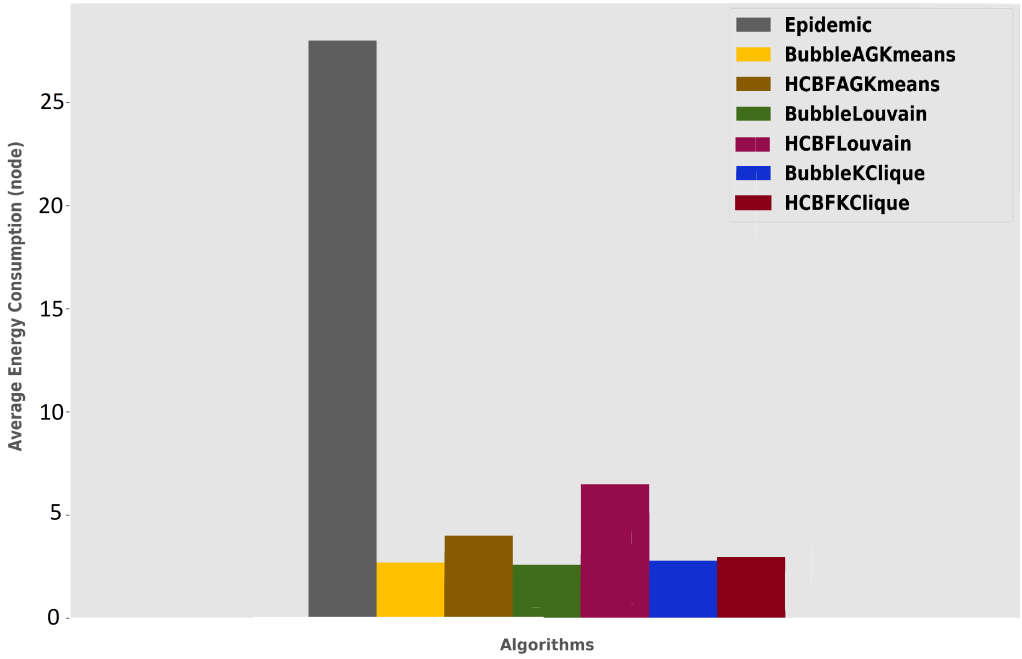
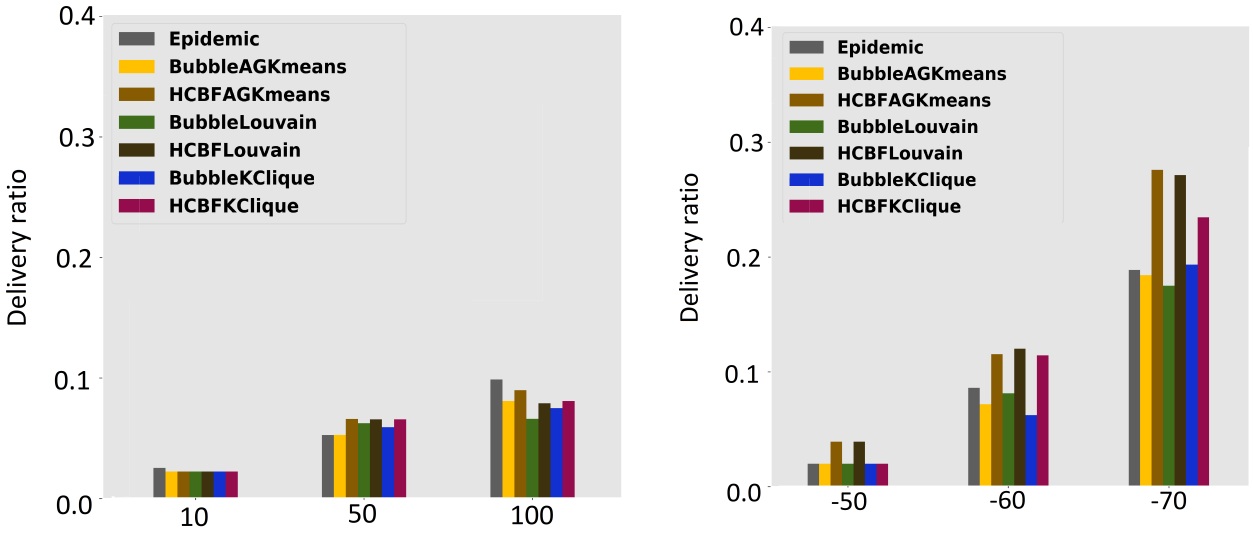


Figure 5.29: Average energy consumption - Limited resources plant and habitat monitoring - 7 days TTL - Uniform Message Generation - Termite

5.3.4 Disaster Relief Scenario with Limited Resources

For disaster relief scenarios, the SHED1 and SHED5 datasets have been used to create a PSN network with dynamic nodes that the source and destination of messages are selected from different clusters. In this scenario, packets would be expired after 7 days and each buffer can store 10 packets. To implement the scenario, it has been considered that some nodes are located in the disaster area and others are located out of the disaster area. Messages are generated in the disaster area and the destinations are out of the area. For implementing a disaster relief scenario in PYDTNSIM, a power function distribution has been used as a message generation parameter. The power function distribution begins by generating one message per duty cycle and decreases the number of messages generated in such that at the end of the simulation for the final 10 duty cycles, only one message is generated. Power function distribution increases the delivery ratio by reducing the number of messages generated at the end of simulation that have a low chance of being delivered.

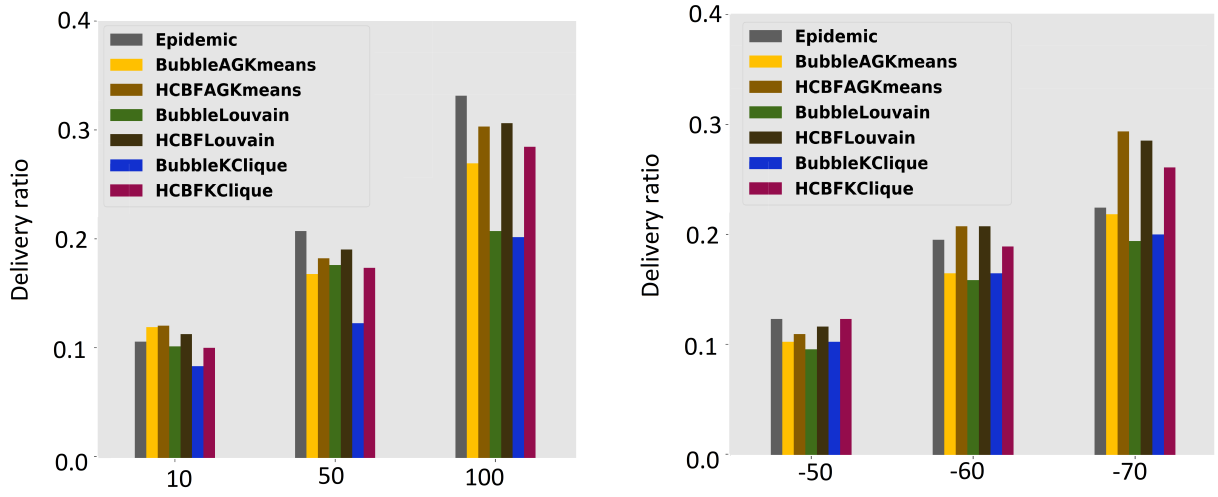
Figures 5.30 and 5.31 demonstrate the delivery ratio for a disaster relief scenario. In comparison to the limited resources, random source and destination scenario, the delivery ratio of disaster relief scenario have been improved. This improvement can be explained by generating fewer messages in the network and especially reducing the remaining messages at the end of the simulation which causes a reduction in message generation and increasing the delivery ratio. Still in all scenarios, the delivery ratio of algorithms that use HCBF is higher than those that use Bubble and the diversity of contacts has a positive effect on the delivery ratio.



(a) GPS

(b) WiFi

Figure 5.30: Delivery ratio SHED1 Limited resources Disaster Relief - 7 days TTL - Power function Message Generation

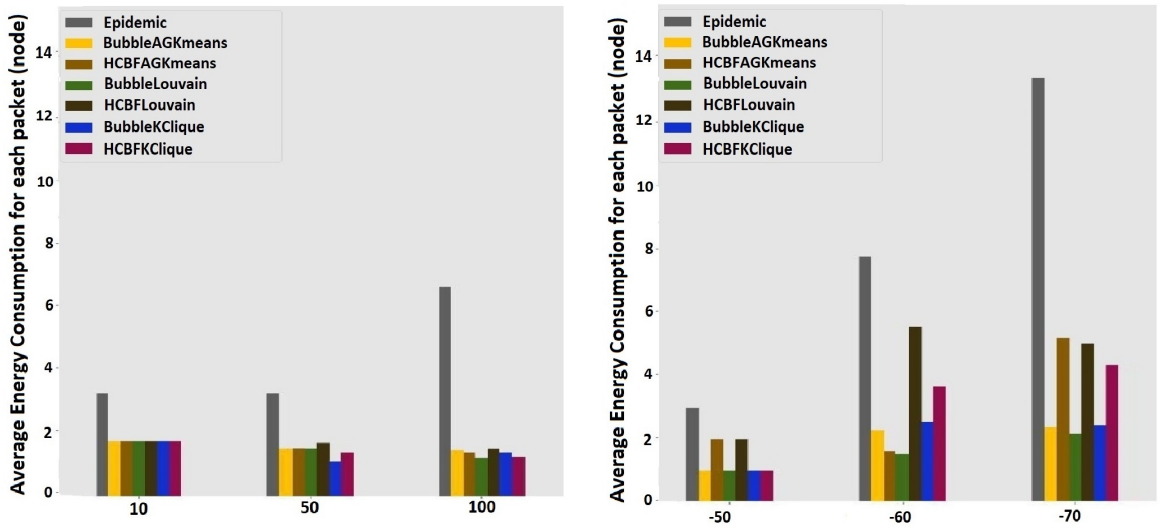


(a) GPS

(b) WiFi

Figure 5.31: Delivery ratio SHED5 Limited resources Disaster Relief - 7 days TTL - Power function Message Generation

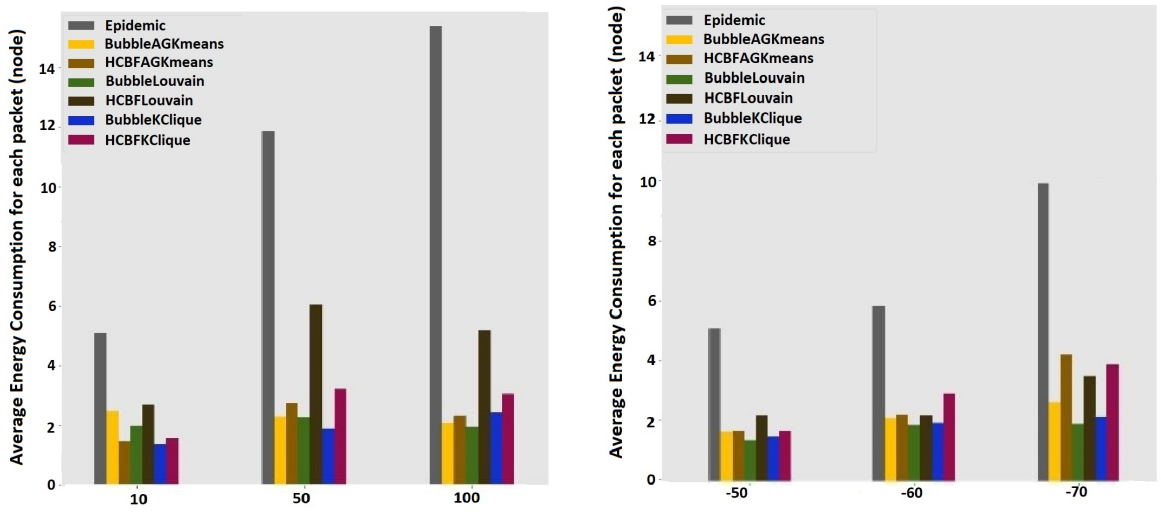
Figures 5.32 and 5.33 demonstrate the average energy consumption for each packet in a disaster relief scenario. In comparison to the random and source and destination, the average energy consumption has increased. The reason for this increase can be explained by the selection of source and destination.



(a) GPS

(b) WiFi

Figure 5.32: Average energy consumption SHED1 Limited resources Disaster Relief - 7 days TTL - Power function Message Generation



(a) GPS

(b) WiFi

Figure 5.33: Average energy consumption SHED5 Limited resources Disaster Relief - 7 days TTL - Power function Message Generation

In random source and destination, the possibility of direct contact among source and destination and the intra-cluster message forwarding decrease the average energy consumption. However, in disaster relief scenarios, the source and destination of packets are selected from different clusters which cause higher average energy consumption. In all scenarios, the average energy consumption for each packet of algorithms that utilize HCBF is higher than Bubble.

5.4 Execution Time of Algorithms

The execution time of the algorithms can represent how quickly the algorithm can be run and how efficiently it uses the resources of the compiling machine [42]. The execution time of algorithms in this thesis is calculated by measuring the start time, and how much time it takes until the program ends. All experiments are run on an Intel Core i5-7200U processor. Figure 5.34 demonstrates the execution time of all algorithms in seconds.

The Epidemic algorithm has the highest execution time. The execution time of the Epidemic algorithm is affected by the high number of copied messages generated and each of these copies require to be routed separately. Therefore, in Epidemic routing algorithms the number of messages in the network is much higher than single-copy algorithms which cause a higher execution time.

Algorithms that use HCBF consume more time for execution than algorithms that use Bubble. In cluster-based routing algorithms, forwarding decisions in each message transmission depend on the values of social factors. The calculation of social factors needs time. As HCBF has four different factors for message forwarding and Bubble has two factors for message forwarding decisions the execution time of HCBF is

higher than Bubble.

Clustering algorithms require different amounts of time for execution. AGKmeans is the worst for clustering as the initial centroid selection and iterations for finding appropriate centroids are time-consuming. KClique is the best in execution time as it only requires one polynomial iteration [52] for finding full cliques in graphs.[3] Louvain algorithm execution time has the highest range of variations which can depict the variation in the number of iterations for forming cluster in different run of the algorithms.

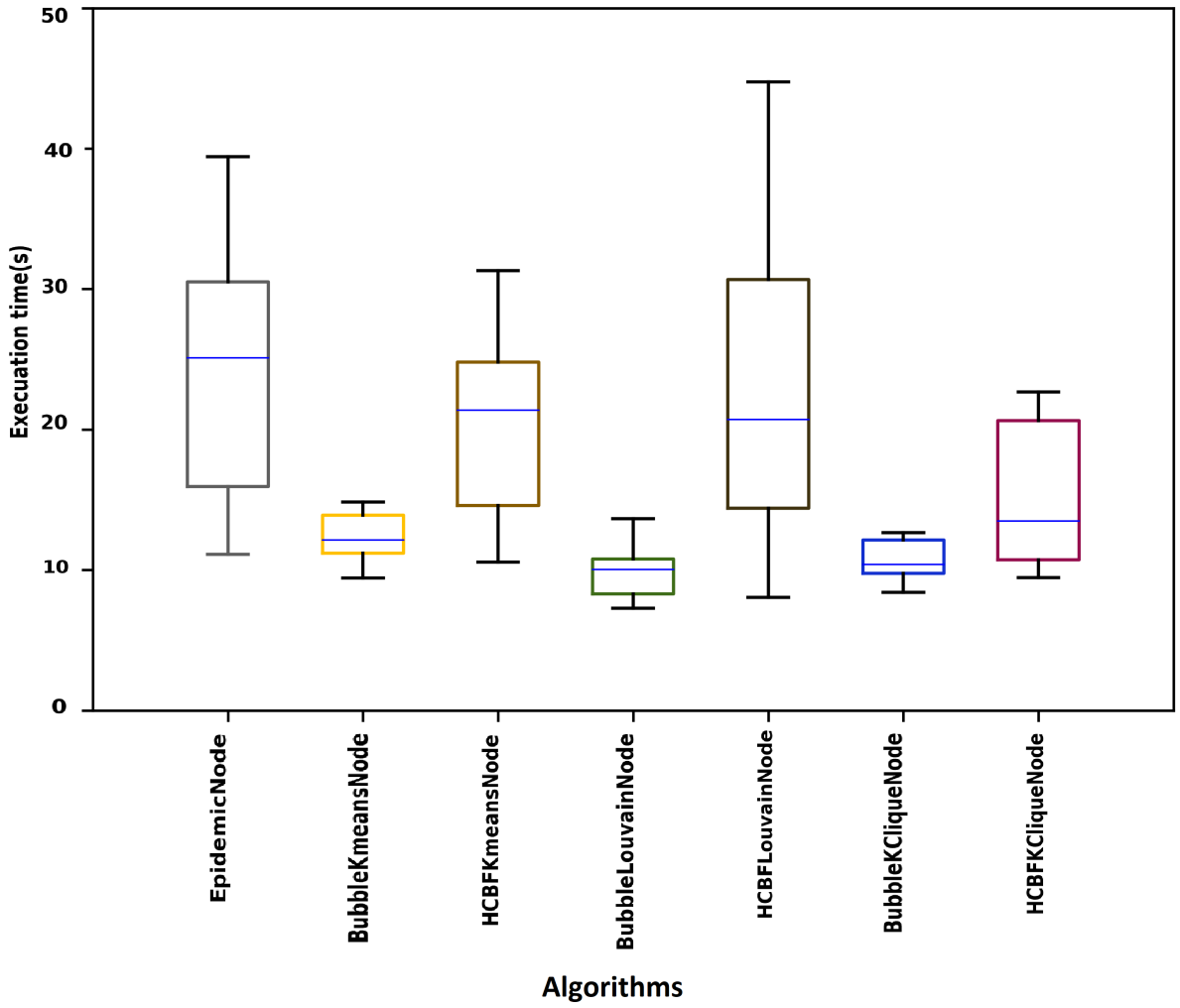


Figure 5.34: Execution time of algorithms(seconds)

5.4.1 Summary

In this chapter, the effect of the transmission range of devices, network resource capacity, the topology of devices, contact stability, clustering algorithms, and message generation rates on the performance metrics such

[3] https://networkx.github.io/documentation/networkx-1.10/reference/generated/networkx.algorithms.community.kclique.k_clique_communities.html

as delivery ratio, energy consumption, execution time, and delay of DTNs algorithms have been investigated.

The results represented that DTNs algorithm performance can be improved by datasets features such the number of contacts, high transmission range of devices, short-lasting, and diverse contacts. The policies of message forwarding can also affect the performance of DTNs algorithms. One unclustered (Epidemic) and two cluster-based routing algorithms(HCBF and Bubble) and three clustering algorithms (KCliques, Louvain, and AGKmeans) were implemented in PYDTNSIM. These algorithms have different performance in different use case scenarios and resource capacities. However, the unclustered algorithm has a good performance in unlimited resources scenarios, cluster-based algorithms have a better performance than unclustered algorithms in limited scenarios. Routing algorithms that use HCBF have a higher delivery ratio than Bubble. The execution time of the algorithms is affected by the high number of messages that require to be forwarded. HCBF forwarding decisions need more calculations than Bubble which increased the execution time of HCBF algorithms and in clustering algorithms, the number of iteration for forming clusters can affect the execution time of the algorithms.

Chapter 6

Conclusion

For efficient routing in human-based DTNs, utilizing the contact patterns of humans can improve the performance of routing algorithms. Previous studies, HCBF and Bubble, proposed different heuristics for inter-clustering and intra-clustering routing algorithms to improve the performance of PSNs algorithms by selecting appropriate carrier nodes for messages.

Not only routing algorithms, but also the topology of devices, contact stability, device transmission range, network resources capacity, clustering algorithm, and message generation rate can affect the performance of PSNs routing algorithms. To determine the effect of clustering techniques on the performance of cluster-based routing algorithms, AGKmeans is proposed in this thesis by using the Kmeans clustering concept. This algorithm is appropriate for datasets in which the participants are dynamic. Initial centroids selection in AGKmeans is not performed randomly; centroids are selected based on their interactions to reduce the execution time and increase the reliability of the algorithm.

Although the network topology of devices, contacts stability and the transmission range of devices are related to the datasets, clustering techniques, variable message generation, and network resources capacity needs to be supported by simulation environments. PYDTNSIM has been extended in this thesis to support cluster-based routing algorithms. PYDTNSIM supports variable message sizes, different source and destination patterns, adjustable message generation and message transmissions tracker. Data analysis of datasets is done to determine the effect of contact stability, network topology, and transmission range of devices on the performance of PSNs.

6.1 Results analysis

The PSN algorithms have been compared for two different capacities for resources: unlimited and constraints. However, by constraining the resources the performance of all routing algorithms reduced, multi-copy algorithms had the highest performance reduction which is the result of the high number of copied messages occupied the buffer space of devices. The evaluation of PSN algorithms has been evaluated against the SHED1 and SHED5 datasets. GPS and WiFi have been used as proxies for contacts. Different methods of contact extraction lead to different network topologies. GPS contacts were extracted using a distance threshold and WiFi contacts extracted based on signal strength recognized by connection to the same WiFi

routers. The WiFi contact topology is composed of several vertex disjoint subgraphs that have a fully connected topology. The GPS contact network usually generates a partially connected topology with a low number of vertex disjoint subgraphs. The performance of PSNs demonstrated that with the same number of contacts, the performance of algorithms applied to WiFi contact topology are higher than GPS. Furthermore, in both WiFi and GPS environments by increasing the transmission range the delivery ratio increased, which is the result of having more contacts and more chances for finding appropriate carrier nodes.

In datasets with the same number of contacts, short-lasting contacts can improve the delivery ratio. Short-lasting contacts can meet more diverse devices which increase the chance of finding appropriate carrier nodes for delivering the messages to the respective destination.

Three clustering algorithms were implemented in PYDTNSIM: KCliques, Louvain, and AGKmeans. KCliques tries to find fully connected subgraphs as clusters. The diversity of nodes contacts would be maximized in each cluster. KClique is not a good algorithm for balancing the number of nodes in each cluster which makes inter-cluster decisions making less useless. Louvain and AGKmeans try to increase the weight of edges in each cluster by maximizing the popularity of nodes. The PYDTNSIM simulation results determined that KClique does perform well compared to Louvain and AGkmeans. In energy consumption, AGKmeans has a better performance in most cases, whereas Louvain acts better in the delivery ratio.

These results show that cluster-based routing algorithms benefit from clustering techniques. Selecting nodes which have more or stronger contacts as carrier nodes can improve the chance of message delivery.

Two message generation rates have been used based on Uniform and Power function distributions. The results demonstrated that power function distribution has a better performance than the uniform one because fewer messages are orphaned at the end of the simulation.

The execution time of the Epidemic algorithm is affected by the high number of copied messages generated. Adding social factors by HCBF for forwarding decisions needs more calculations than Bubble which increased the execution time of HCBf algorithms. In clustering algorithms, the number of iteration for forming clusters can affect the execution time of the algorithms.

6.2 Future Work

Our research has several shortcomings that could be addressed in future works.

Clustering algorithms: As routing by clustering techniques can improve the performance of DTNs, finding appropriate clustering algorithms is important. For example, KClique which has been used in Bubble can improve the performance of routing but this improvement is less than what can be achieved using AGKmeans and Louvain, likely because for dense networks KCLique detects fewer clusters. The Louvain algorithm is stable and does not need the number of clusters to be specified, however, in some metrics such as energy consumption AGKmeans outperforms Louvain. AGKmeans demonstrate good performance but require a specification of k. Adding an assignment and update phase could improve the performance of this

algorithm and also remove the strict dependency of the algorithm on k.

In the real implementation of DTNs, nodes need to have a list of clusters and their members, along with the CBC, NCF, and UI for the other nodes. A purely decentralized algorithm would be possible, but have substantially different properties, because using only local information may result in nodes thinking they have different cluster members.

PYDTNSIM: Although PYDTNSIM is modular and can be easily extended for developing and supporting other routing and clustering algorithms, PYDTNSIM does not have a strong GUI for the usage of new users. PYDTNSIM have lots explanation as comments which make it understandable but still, it is useful for users who know Python. Adding GUI for PYDTNSIM can make it usable for users who are not familiar with Python. Currently, PYDTNSIM does not support multicast message forwarding that each message has a set of destination nodes. This could be a fairly simple enhancement to PYDTNSIM, just each message needs an array of destinations. If the message encountered one member of that set, the the message would be counted as a delivered message.

Use Case Scenarios and DTN algorithms: In this thesis, three use case scenarios of DTNs have been evaluated. DTNs have lots of various use cases such as military or transportation improvements that still can be evaluated by PYDTNSIM and cluster-based routing algorithms designed for DTN. The results of PYDTNSIM determined cluster-based routing algorithms cannot gain a high performance in static datasets. Evaluating PYDTNSIM with other use cases can determine their efficiency in different scenarios. For example, use cases where dynamic and static nodes generate differing amounts of traffic can give a better understanding of DTN's performance in static datasets. The priority of messages encoded by the sender can also be evaluated in future studies because currently the priority is only based on expiry time in PYDTNSIM. Furthermore, comparing different metrics such as delivery time of the forward path to the backward can demonstrate a better view of DTN.

Network resources and Message sizes: In this thesis, the scenarios have been evaluated with both unlimited and limited network resources. The limited network resources considered fixed in all scenarios. Evaluating DTNs algorithms with variable network resources such as different buffer space, bandwidth with different message sizes can represent the effect of these on DTNs algorithms performance.

Bibliography

[1] Agarwal, R. K., M. Mathuria, and M. Sharma (2017), Study of Routing Algorithms in DTN Enabled Vehicular Ad-Hoc Network, *International Journal of Computer Applications*, pp. 1–6.

[2] Ahsanullah, M., and A. L. Kabir (1974), A characterization of the power function distribution, *The Canadian Journal of Statistics/La Revue Canadienne de Statistique*, pp. 95–98.

[3] Babu, S., G. Jain, and B. Manoj (2017), Urban Delay Tolerant Network Simulator (UDTNSim v0. 1), *arXiv preprint*, pp. 1–40.

[4] Barber, M. N., N. Nakanishi, B. Ninham, and B. Ninham (1970), Random and restricted walks: Theory and Applications, *CRC Press*, pp. 5–13.

[5] Blondel, V. D., J.-L. Guillaume, R. Lambiotte, and E. Lefebvre (2008), Fast unfolding of communities in large networks, *Journal of Statistical Mechanics: Theory and Experiment*, pp. 1–12.

[6] Bruno, R., N. Santos, and P. Ferreira (2015), Termite: Emulation testbed for encounter networks, in *Proceedings of the 12th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services on 12th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, pp. 31–40, Coimbra, Portugal.

[7] Bulusu, N., J. Heidemann, D. Estrin, et al. (2000), GPS-less low-cost outdoor localization for very small devices, *IEEE Personal Communications*, pp. 28–34.

[8] Bulut, E., and B. K. Szymanski (2012), Exploiting friendship relations for efficient routing in mobile social networks, *IEEE Transactions on Parallel and Distributed Systems*, pp. 2254–2265.

[9] Burgess, J., B. Gallagher, D. D. Jensen, and B. N. Levine (2006), Maxprop: Routing for vehicle-based disruption-tolerant networks, in *Proceedings of the INFOCOM*, pp. 1688–1698, Barcelona, Spain.

[10] Caini, C., H. Cruickshank, S. Farrell, and M. Marchese (2011), Delay-and disruption-tolerant networking (DTN): an alternative solution for future satellite networking applications, in *Proceedings of the IEEE*, pp. 1980–1997.

[11] Cao, Y., and Z. Sun (2012), Routing in delay/disruption tolerant networks: A taxonomy, survey and challenges, *IEEE Communications Surveys & Tutorials*, pp. 654–677.

[12] Carettoni, L., C. Merloni, and S. Zanero (2007), Studying bluetooth malware propagation: The bluebag project, *IEEE Security & Privacy*, pp. 17–25.

[13] Chen, K., and H. Shen (2012), SMART: Lightweight distributed social map based routing in delay tolerant networks, in *Proceedings of the 2012 20th IEEE International Conference on Network Protocols (ICNP)*, pp. 1–10, Piscataway, NJ.

[14] Daly, E. M., and M. Haahr (2007), Social network analysis for routing in disconnected delay-tolerant manets, in *Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*, pp. 32–40, Montreal, Quebec, Canada.

[15] Deza, M. M., and E. Deza (2009), Encyclopedia of distances, in *Encyclopedia of Distances*, pp. 1–583, Springer.

[16] Dijkstra, E. W. (1959), A note on two problems in connexion with graphs, *Numerische Mathematik*, pp. 269–271.

[17] Dong, Q., and W. Dargie (2012), Evaluation of the reliability of RSSI for indoor localization, in *2012 International Conference on Wireless Communications in Underground and Confined Areas*, pp. 1–6, Clermont-Ferrand, France.

[18] Duan, D., Y. Li, R. Li, and Z. Lu (2012), Incremental K-clique clustering in dynamic social networks, *Artificial Intelligence Review*, pp. 129–147.

[19] Edachery, J., A. Sen, and F. J. Brandenburg (1999), Graph clustering using distance-k cliques, in *International Symposium on Graph Drawing*, pp. 98–106, Springer, Berlin, Germany.

[20] Eisenman, S. B., E. Miluzzo, N. D. Lane, R. A. Peterson, G.-S. Ahn, and A. T. Campbell (2009), BikeNet: A mobile sensing system for cyclist experience mapping, *ACM Transactions on Sensor Networks (TOSN)*, pp. 1–6.

[21] Fall, K., G. Iannaccone, J. Kannan, F. Silveira, and N. Taft (2010), A disruption-tolerant architecture for secure and efficient disaster response communications, in *Proceedings of the 7th International ISCRAM Conference*, pp. 1–6, Seattle, WA.

[22] Farrell, S., V. Cahill, D. Geraghty, I. Humphreys, and P. McDonald (2006), When TCP breaks: Delay- and disruption-tolerant networking, *IEEE Internet Computing*, pp. 72–78.

[23] Fida, M.-R., M. Ali, and A. S. Arsalaan (2011), DTN-RSim: An event-based routing simulator for dtn, in *Proceedings of the International Conference on Computer Networks and Information Technology*, pp. 99–104, Bara Gali, Pakistan.

[24] Fraire, J. A., P. G. Madoery, F. D. Raverta, J. M. Finochietto, and R. Velazco (2017), DtnSim: Bridging the Gap between Simulation and Implementation of Space-Terrestrial DTNs, in *Proceedings of the 6th International Conference on Space Mission Challenges for Information Technology (SMC-IT)*, pp. 120–123, Madrid, Spain.

[25] Freeman, L. C. (1977), A set of measures of centrality based on betweenness, *Sociometry*, pp. 35–41.

[26] Gibbs, M. (2008), Product review: Metageek wi-spy 2.4 x, *Network World Canada*.

[27] Girvan, M., and M. E. Newman (2002), Community structure in social and biological networks, *Proceedings of the National Academy of Sciences*, pp. 7821–7826.

[28] Gunda, P. K., L. Ravindranath, C. A. Thekkath, Y. Yu, and L. Zhuang (2010), Nectar: Automatic management of data and computation in datacenters., in *Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation*, pp. 75–88, Vancouver, Canada.

[29] Guo, S., M. H. Falaki, E. A. Oliver, S. Ur Rahman, A. Seth, M. A. Zaharia, and S. Keshav (2007), Very low-cost internet access using KioskNet, *ACM SIGCOMM Computer Communication Review*, pp. 95–100.

[30] Gupta, A. K., I. Bhattacharya, P. S. Banerjee, J. K. Mandal, and A. Mukherjee (2016), Dirmove: direction of movement based routing in DTN architecture for post-disaster scenario, *Wireless Networks*, pp. 723–740.

[31] Hartenstein, H., and K. Laberteaux (2010), VANET: Vehicular Applications and inter-Networking Technologies, pp. 1–6, Wiley Online Library.

[32] Hashemian, M. S., K. G. Stanley, D. L. Knowles, J. Calver, and N. D. Osgood (2012), Human network data collection in the wild: the epidemiological utility of micro-contact and location data, in *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium*, pp. 255–264, Miami, FL.

[33] Helgason, Ó. R., and K. V. Jónsson (2008), Opportunistic networking in OMNeT++, in *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops*, pp. 82–90, Marseille, France.

[34] Hui, P., A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot (2005), Pocket switched networks and human mobility in conference environments, in *Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-tolerant Networking*, pp. 244–251, New York, NY.

[35] Hui, P., J. Crowcroft, and E. Yoneki (2010), Bubble rap: Social-based forwarding in delay-tolerant networks, *IEEE Transactions on Mobile Computing*, pp. 1576–1589.

[36] Jackson, J. (2005), The interplanetary internet [networked space communications], *IEEE spectrum*, pp. 30–35.

[37] Jain, S., K. Fall, and R. Patra (2004), Routing in a Delay Tolerant Network, in *Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pp. 145–158, Portland, OR.

[38] Jonson, T., J. Pezeshki, V. Chao, K. Smith, and J. Fazio (2008), Application of delay tolerant networking (DTN) in airborne networks, in *Proceedings of the MILCOM IEEE Military Communications Conference*, pp. 1–7, San Diego, CA.

[39] Kang, H., S. H. Ahmed, D. Kim, and Y.-S. Chung (2015), Routing protocols for vehicular delay tolerant networks: a survey, *International Journal of Distributed Sensor Networks*, pp. 1–9.

[40] Keränen, A., J. Ott, and T. Kärkkäinen (2009), The ONE Simulator for DTN Protocol Evaluation, in *Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, pp. 55–64, Rome, Italy.

[41] Kernighan, B. W., and S. Lin (1970), An efficient heuristic procedure for partitioning graphs, *Bell System Technical Journal*, pp. 291–307.

[42] Knuth, D. E. (1970), The analysis of algorithms, in *Actes du congres international des Mathématiciens*, pp. 49–62.

[43] Kowsalya, M., and M. Soranamageswari (2014), A survey on delay tolerant network, *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, pp. 2450–2455.

[44] Lindgren, A., and P. Hui (2011), Extremecom: to boldly go where no one has gone before, *ACM SIGCOMM Computer Communication Review*, pp. 54–59.

[45] Lindgren, A., A. Doria, and O. Schelen (2004), Probabilistic routing in intermittently connected networks, in *Proceedings of the International Workshop on Service Assurance with Partial and Intermittent Resources*, pp. 239–254, Fortaleza, Brazil.

[46] Lindgren, A., A. Doria, J. Lindblom, and M. Ek (2008), Networking in the land of northern lights: two years of experiences from DTN system deployments, in *Proceedings of the 2008 ACM Workshop on Wireless Networks and Systems for Developing Regions*, pp. 1–8, San Francisco, CA.

[47] Liu, M., Y. Yang, and Z. Qin (2011), A survey of routing protocols and simulations in delay-tolerant networks, in *International Conference on Wireless Algorithms, Systems, and Applications*, pp. 243–253, Berlin, Germany.

[48] Lloyd, S. (1982), Least squares quantization in PCM, *IEEE transactions on information theory*, pp. 129–137.

[49] Luk, R., M. Zaharia, M. Ho, B. Levine, and P. M. Aoki (2009), ICTD for healthcare in Ghana: two parallel case studies, in *Proceedings of the 2009 International Conference on Information and Communication Technologies and Development (ICTD)*, pp. 118–128, Doha, Qatar.

[50] Newman, M. E. (2003), The structure and function of complex networks, *SIAM review*, pp. 167–256.

[51] Ochiai, H., H. Ishizuka, Y. Kawakami, and H. Esaki (2011), A DTN-based sensor data gathering for agricultural applications, *IEEE Sensors Journal*, pp. 2861–2868.

[52] Palla, G., I. Derényi, I. Farkas, and T. Vicsek (2005), Uncovering the overlapping community structure of complex networks in nature and society, *Nature*, pp. 1–10.

[53] Pentland, A., R. Fletcher, and A. Hasson (2004), Daknet: Rethinking connectivity in developing nations, *Computer*, pp. 78–83.

[54] Qian, W., K. G. Stanley, and N. D. Osgood (2013), The impact of spatial resolution and representation on human mobility predictability, in *Proceedings of the International Symposium on Web and Wireless Geographical Information Systems*, pp. 25–40, Berlin, Germany.

[55] Rasul, K. (2013), Enhanced Community-Based Routing for Low-Capacity Pocket Switched Networks, Ph.D. thesis, University of Saskatchewan.

[56] Rasul, K., S. A. Chowdhury, D. Makaroff, and K. Stanley (2014), Community-based forwarding for low-capacity pocket switched networks, in *Proceedings of the 17th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pp. 249–257, Montreal, Canada.

[57] Rasul, K., D. Makaroff, and K. G. Stanley (2015), Hybrid community-based forwarding: A complete energy efficient algorithm for pocket switched networks, in *2015 IEEE 40th Local Computer Networks Conference Workshops (LCN Workshops)*, pp. 760–768, Clearwater Beach, FL.

[58] Ritchel, M. (2011), Egypt cuts off most internet and cell service, *The New York Times*.

[59] Royer, E. M., and C.-K. Toh (1999), A review of current routing protocols for ad hoc mobile wireless networks, *IEEE personal communications*, pp. 46–55.

[60] Ryu, J., L. Ying, and S. Shakkottai (2010), Back-pressure routing for intermittently connected networks, in *Proceedings of the 2010 IEEE INFOCOM*, pp. 1–5, San Diego, CA.

[61] Schildt, S., J. Morgenroth, W.-B. Pöttner, and L. Wolf (2011), IBR-DTN: A lightweight, modular and highly portable bundle protocol implementation, *Electronic Communications of the EASST*, pp. 1–10.

[62] Scott, J. (1988), Social network analysis, *Sociology*, pp. 109–127.

[63] Seye, M. R., M. Diallo, B. Gueye, and C. Cambier (2019), COWShED: Communication Within White Spots for Breeders, in *Proceedings of the 2019 22nd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, pp. 236–238, Paris, France.

[64] Shah, H., and Y. P. Kosta (2010), Evolution of routing techniques, routing protocols and routing efficiencies for delay tolerant network, *International Journal of Computer Applications (Special issue on MANETs)*, pp. 46–53.

[65] Spaho, E., K. Bylykbashi, L. Barolli, and M. Takizawa (2017), Routing in a DTN: performance evaluation for random waypoint and steady state random waypoint using ns3 simulator, in *Proceedings of the International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*, pp. 133–141, Barcelona, Spain.

[66] Spanakis, E. G., and A. G. Voyiatzis (2012), DAPHNE: A disruption-tolerant application proxy for e-health network environments, in *International Conference on Wireless Mobile Communication and Healthcare*, pp. 88–95, Paris, France.

[67] Spyropoulos, T., K. Psounis, and C. S. Raghavendra (2005), Spray and wait: an efficient routing scheme for intermittently connected mobile networks, in *Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-tolerant Networking*, pp. 252–259, Philadelphia, PA.

[68] Spyropoulos, T., K. Psounis, and C. S. Raghavendra (2008), Efficient routing in intermittently connected mobile networks: The single-copy case, *IEEE/ACM Transactions on Networking*, pp. 63–76.

[69] Sun, W., C. Liu, and D. Wang (2011), On delay-tolerant networking and its application, in *International Conference on Computer Science and Information Technology (ICCSIT)*, pp. 238–244, Penang, Malaysia.

[70] Tan, P.-N., M. Steinbach, and V. Kumar (2006), Cluster analysis: basic concepts and algorithms, *Introduction to Data Mining*, pp. 487–568.

[71] Tasfe, M., and A. Chakrabarty (2017), Gossip: A social interest based routing algorithm for pocket switched network, in *Proceedings of the 2017 20th International Conference of Computer and Information Technology (ICCIT)*, pp. 1–6, Dhaka, Bangladesh.

[72] Tovar, A., T. Friesen, K. Ferens, and B. McLeod (2010), A DTN wireless sensor network for wildlife habitat monitoring, in *Proceedings of the Annual IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, pp. 1–5, Calgary, Canada.

[73] Uddin, M. Y. S., D. M. Nicol, T. F. Abdelzaher, and R. H. Kravets (2009), A post-disaster mobility model for delay tolerant networking, in *Proceedings of the Winter Simulation Conference*, pp. 2785–2796, Austin, Texas.

[74] Vahdat, A., and D. Becker (2000), Epidemic routing for partially connected ad hoc networks, *Technical Report CS-200006, Duke University.*

[75] Varga, A., and R. Hornig (2008), An overview of the OMNeT++ simulation environment, in *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops*, pp. 60–70, Marseille, France.

[76] Velásquez-Villada, C., and Y. Donoso (2016), Delay/disruption tolerant network-based message forwarding for a river pollution monitoring wireless sensor network application, in *Sensors*, vol. 16, pp. 96–110, Multidisciplinary Digital Publishing Institute.

[77] Voyiatzis, A. G. (2012), A survey of delay-and disruption-tolerant networking applications, *Citeseer.*

[78] Wood, L., W. Ivancic, W. Eddy, D. Stewart, and J. Northam (2012), Investigating operation of the Internet in orbit: Five years of collaboration around CLEO, *arXiv preprint arXiv:1204.3261*, pp. 10–11.

[79] Yu, D., and Y.-B. Ko (2009), FFRDV: fastest-ferry routing in dtn-enabled vehicular ad hoc networks, in *2009 11th International Conference on Advanced Communication Technology*, pp. 1410–1414, Gangwon-Do, South Korea.

[80] Zhang, P., C. M. Sadler, S. A. Lyon, and M. Martonosi (2004), Hardware design experiences in ZebraNet, in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, pp. 227–238, Baltimore, MD.

Appendix A

PYDTNSIM Installation and Running

To install PYDTNSIM, simply use the following command:
pip install git + https : //git.cs.usask.ca/discus/pydtnsim.git
For Development of PYDTNSIM clone the repository via ssh by the following command:
git clone git@git.cs.usask.ca : discus/pydtnsim.git
or via https:
git clone https : //git.cs.usask.ca/discus/pydtnsim.git
PYDTNSIM can be run on Python integrated development environment (IDE) such as PyCharm. PyCharm can be downloaded from https://www.jetbrains.com/pycharm/download/. The community version is open-source and can be used for PYDTNSIM running. After installation of PyCharm, open pydtn-develop on it. Then install the package and its dependencies:
cd pydtn
pip install − −editable.
PYDTNSIM can be run with Python 3.4 and 3.6. To run PYDTNSIM three following packages with the specific versions are required:
networkx==1.11
simpy==3.0.10
python-louvain==0.8
To run the PYDTNSIM for a specific dataset, the contact dataset needs to be saved in the data folder of pydtn-develop. A metadata file is required in the same folder which contains information such as the duration of the dataset, the number of participants. For running the PYDTNSIM type the following command in the terminal:
python examples\shed.py data\filename

Appendix B

Termite Installation and Running

To install termite download termite from the following link: [1]
http://www.gsd.inesc-id.pt/ wiki/courses/cmu1516/lab04/Termite-Cli-20160329.tgz
Open a terminal window and navigate to directory $Termite - Cli$
Set the environment variables
To execute the Termite client, run *termite.bat*
Then download android studio from the following link and install it:
https://developer.android.com/studio
Download the PeerScanner app and decompress it on a local directory. Open PeerScanner on Android Studio.
http://www.gsd.inesc-id.pt/ wiki/courses/cmu1516/lab04/Termite-WifiP2P-PeerScanner-20160329.tgz
On Termite, create a virtual devices by the following command:
newdevice A
Run Virtual Device on Android studio and set your parameters such as signal strength, latitude and longitude of device
Assign network address to each emulator individually using the following command:
assignaddr
Bind the emulator to the virtual device A using the following command:
binddevice

[1] https://nuno-santos.github.io/termite/installation.html